ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:
INGENIERO EN INFORMÁTICA

Título del proyecto:

"Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica"

Oscar Echeverría Esparza

Tutor: José Javier Astrain Escola

Pamplona, 22 de Febrero de 2013

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

AGRADECIMIENTOS

Quiero aprovechar estas líneas para agradecer a todas las personas que me han ayudado, en concreto a mi tutor José Javier Astrain que siempre ha estado disponible para responder a cualquier duda o proponer soluciones a cualquier problema surgido.

Y en especial, quiero mostrar mi más sincero agradecimiento a Óscar Araiztegui Ascunce, Ingeniero Técnico del Servicio de Obras, Equipamiento y Mantenimiento del Servicio Navarro de Salud-Osasunbidea, por su alta disponibilidad, por proporcionarnos la información necesaria y sobre todo, destacar su gran involucración en este proyecto.

Gracias por haberme dado la posibilidad de colaborar en este bonito proyecto, a la par que difícil, y poder contribuir con mi granito de arena en la mejora del entorno actual en el que nos movemos.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONTENIDO

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INTRODUCCIÓN

Capítulo de introducción al proyecto. En él se describe brevemente el problema, el estado del arte, los objetivos que marcan la realización del proyecto, la propuesta final de desarrollo y la metodología a seguir.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

1 INTRODUCCIÓN

1.1 PROBLEMA A RESOLVER

El Servicio Navarro de Salud-Osasunbidea (SNS-O), el cual es responsable de numerosos hospitales y centros de salud realiza un enorme consumo en sus instalaciones.

Por ello necesita que sus empleados realicen un esfuerzo para poder analizar cada una de las facturas que llegan, y posterior generación de los informes necesarios que les ayuden al cálculo de la evaluación de los gastos realizados, para en un futuro ajustar sus tarifas y contratos con el fin de gastar lo mínimo posible.

Por ello se creó la necesidad del desarrollo de un programa que automatizase dichas tareas comentadas anteriormente.

El presente proyecto tiene como objetivo la continuación del desarrollo de un programa informático iniciado por Sandra Huguet [1] e Igor Trébol [2] en 2009.

Desde entonces se ha producido algún cambio, como por ejemplo el cambio de la compañía de suministro de energía contratada por el Servicio Navarro de Salud, de Endesa a Unión Fenosa. Esta nueva empresa cambia de formato en la información de las facturas, pasando de XML (que proporcionaba Endesa) a Excel, con lo que conlleva un cambio en la aplicación. Además esta nueva compañía no desvela información sobre los consumos de los cuartos horarios, por lo que no se puede estimar el cálculo de las facturas y verificar que coinciden con los proporcionados por la compañía, sólo se puede verificar si los cálculos realizados en el Excel son correctos.

El proyecto consiste en una interfaz gráfica que permite la introducción y consulta de los datos manejados por el sistema y almacenados en una base de datos. La idea es mejorar dicho proyecto iniciado y adaptarlo a los cambios surgidos.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

1.2 ESTADO DEL ARTE

Hasta Julio de 2009 la empresa comercializadora de la electricidad era la distribuidora a su vez, sin embargo, desde entonces entró un nuevo modelo de suministro de electricidad, obligando a que la electricidad sólo pudiera ser comercializada por aquellas empresas dedicadas únicamente a la comercialización, cuyas consecuencias fueron la multiplicación de las ofertas y tarifas, haciendo así mucho más difícil la elección de la más adecuada que se ajuste mejor las necesidades de los usuarios.

Debido a este cambio y una serie de motivos más, tales como la particular estructura que posee el Servicio Navarro de Salud – Osasunbidea y los constantes cambios a los que está sometido por las distintas legislaturas y cambios de gobierno (al ser un organismo público), no existe en el mercado una herramienta adecuada para adaptarla a sus necesidades.
Además, actualmente no existe ninguna legislación que obligue a presentar la información de las facturas en un tipo de formato específico, por lo cual cada compañía es libre a la hora de elegir cómo presenta dicha información.

Además, se ha descubierto que en algún caso hay errores en el cobro de las facturas (no coincide la potencia contratada con la facturada, no coinciden los precios de la tarifa contratada con los precios de la facturada, etc.) por lo que se requiere un sistema capaz de chequear que la energía contratada se corresponda con la facturada para que en el caso de que exista algún tipo de error se puedan realizar las reclamaciones necesarias.

Por todo esto, el departamento de Obras del Servicio Navarro de Salud – Osasunbidea, en colaboración con el Departamento de Ingeniería Matemática e Informática de la Universidad Pública de Navarra han creído conveniente continuar con el desarrollo del proyecto iniciado ya en 2009 adaptándolo a los cambios surgidos y añadiendo más funcionalidades que se crean útiles.

Cabe indicar que en una fase previa del proyecto que ya se ha creó la Base de Datos Relacional en Microsoft SQL Server 2008, en la cual se mantendrá almacenada la información utilizada para la gestión [1], y también se creó el interfaz necesario para la manipulación de la información [2].

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

1.3 OBJETIVOS

Los objetivos que se pretenden conseguir con este proyecto, definidos con la ayuda del SNS-O, son los siguientes:

- Permitir la integración de la información relacionada con los contratos de suministro eléctrico de todos los centros del SNS-O.

- Permitir el control centralizado y distribuido de la facturación.

- Permitir la reducción del gasto a través del análisis de los consumos y las tarifas.

- Permitir el trabajo en red, tanto para la introducción de datos como para la consulta y administración de la herramienta.

- Permitir un enlace automatizado de acceso al sistema de información desde la pasarela de registro electrónico de la empresa suministradora.

1.4 SOLUCIÓN PROPUESTA

Teniendo en cuenta las necesidades de acceso desde distintos puntos, se propuso una aplicación web, que permitiera una interfaz sencilla y accesible con el modelo de datos ya existente como forma de ayuda en la contratación al Servicio Navarro de Salud-Osasunbidea (SNS - O).

Como el SNS – O utiliza herramientas basadas en las tecnologías de Microsoft, la información estará almacenada en una base de datos Microsoft SQL Server 2008, y el desarrollo de la interfaz se efectuará en Microsoft Silverlight 4.

Además, se trata de un prototipo para permitir a los responsables del SNS – O evaluar su funcionalidad y proponer las mejoras que crean convenientes en un futuro.

También se deberá tener en cuenta que han existido problemas en cuanto a los datos suministrados por las compañías eléctricas ya que con Endesa los cálculos efectuados sobre los consumos y los importes facturados no coinciden y Unión Fenosa no proporciona los consumos por cuartos horarios, lo cual impide realizar el cálculo que permita verificar el cobro exacto de los consumos realizados.

Ante el silencio sobre este tema de las compañías, se propone que el prototipo pueda ser fácilmente modificado una vez obtenidos los datos correctos y solucionados estos problemas.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

1.5 METODOLOGÍA

La metodología que se va a seguir para el desarrollo del proyecto es la misma que se ha seguido durante el desarrollo anterior, el proceso unificado de desarrollo de software (RUP-UP).

Esta metodología permite utilizar un ciclo de vida iterativo: así nos va a permitir un desarrollo incremental de la aplicación.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS

Capítulo de análisis de requisitos del sistema, en él se presentan los requisitos funcionales modelados y siguiendo la metodología de casos de uso, así como los requisitos no funcionales

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

2 ANÁLISIS

2.1 ANÁLISIS DE REQUISITOS

Basándonos en el documento llamado "Desarrollo de una Herramienta Informática para la Gestión del Suministro Eléctrico" proporcionado por el SNS – O y tras varias reuniones con el personal responsable se han determinado los requisitos necesarios que a continuación se describen.

2.1.1 REQUISITOS FUNCIONALES

Como punto inicial, el SNS – O requiere un sistema dividido en 3 módulos, distinguiendo diferentes posibilidades de acceso al sistema. La estructura básica puede observarse en el siguiente diagrama:

Figura 1: Estructura Funcional

Cada módulo será accedido de forma independiente por usuarios distintos con distintos privilegios. Tras otras consideraciones se acuerda la división de los usuarios en 3 perfiles o permisos:

- Usuario Administrador.
- Usuario de Datos.
- Usuario de Consulta.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MÓDULO DE ADMINISTRACIÓN

Consistirá en administrar los diferentes usuarios y sus accesos al sistema, gestionando perfiles y dando los permisos requeridos. Podrá crear los usuarios así como los Roles que cada uno pueda tener.

Asimismo, asignará a cada Rol los CUPS o Código Universal de Punto de Suministro, es decir, los puntos de consumo de energía, a los que tiene acceso.

También se encargará de una configuración correcta del entorno, como la introducción en el sistema de los días festivos o los precios generales de la energía registrados en el BOE.

Además se encargará de realizar una copia de seguridad de la base de datos cada vez que se crea conveniente o cada vez que se introduzcan una gran cantidad de datos.

MÓDULO DE DATOS

Módulo encargado de la introducción en el sistema de los datos necesarios para el módulo de Consulta. A pesar de no estar incluido en el módulo de Administración, es el usuario Administrador el único que puede introducir la mayoría de los datos, para evitar posibles problemas.

El usuario de Datos queda relegado a la introducción de facturas, además de a la posibilidad de obtener las consultas e informes que sean necesarias.

BLOQUE DE ORGANIZACIÓN

Tiene como objetivo la introducción de los CUPS y la estructura organizativa del Servicio Navarro de Salud, generando las asociaciones entre las estructuras y los CUPS.

Actualmente los CUPS están asociados a determinados Centros que, a su vez, están integrados en determinadas Áreas que, a su vez, pertenecen al Servicio Navarro de Salud, pero como ya se ha dicho anteriormente, la estructura del SNS – O puede variar en el tiempo. Por ello el sistema debe proporcionar la posibilidad de reorganizar centros y áreas.

En resumen, este bloque debe permitir la creación, modificación y baja de CUPS y la creación y modificación de Organizaciones, pero también debe dar la opción de reestructurar todo el sistema organizativo, asignando, quitando y modificando las asociaciones entre los distintos niveles y de los niveles inferiores con los CUPS.

Tras analizar la situación actual y para dotar al sistema de la mayor flexibilidad, dentro de una visión realista de los posibles cambios, se decide una estructura en cuatro niveles. Cada nivel estaría asociado con los inmediatamente inferiores hasta llegar al nivel 4, que mantendría los CUPS.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

No existe ninguna restricción en cuanto a si un nivel o CUPS puede pertenecer a varios niveles superiores, ni sobre el número de niveles inferiores que pueden llegar a existir. Un ejemplo de una posible organización tendría un aspecto como el que se recoge en el siguiente esquema:

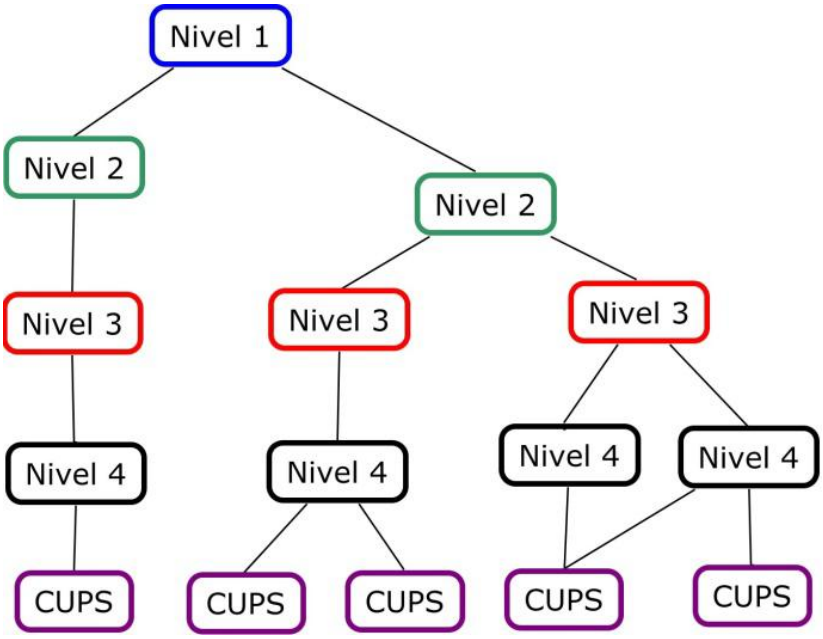


Figura 2: Ejemplo de Estructura Organizativa

En cualquier caso, el sistema deberá mantener un histórico de todas las asociaciones que han existido en el tiempo, para la correcta presentación de informes y consultas.

BLOQUE TARIFAS

El bloque de Tarifas debe permitir la introducción en el sistema de las tarifas que se aplicarán a los contratos que se hayan contratado para cada CUPS, manteniendo información sobre niveles de tensión, escalones de potencia, precios del término de potencia, del término de energía activa y del término de energía reactiva que se aplican en la facturación.

Para adaptarse a los posibles cambios, el sistema debe poder generar las modificaciones que sean necesarias sobre una tarifa sin sobrescribir la misma, ya que para hacer los cálculos de datos anteriores al cambio deberá existir un histórico de todas las existentes.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

También habrá que tener en consideración que las tarifas pueden variar de precio en distintos periodos, de hecho cambian de precio cada entrada de año, además de cada trimestre normalmente, almacenando esa información de manera coherente para los cálculos. En el momento que el usuario quiere modificar algún precio tendrá la posibilidad de ver la evolución del precio de esa tarifa en el tiempo.

Existen también unas tarifas especiales que habrá que tener en cuenta, las Tarifas de Último Recurso (en adelante TUR), tarifas especiales de precio único para todos los consumidores fijadas por el gobierno. Sólo se puede contratar una TUR para una potencia inferior a 10 Kw.

BLOQUE CONTRATOS

La aplicación deberá permitir el alta, baja y modificación de contratos, indicando las condiciones del mismo. Del mismo modo, deberá ser posible asociar un contrato a un determinado CUPS, así como indicar la tarifa aplicable de entre las existentes.

Del mismo modo, deberá mantenerse un histórico, por lo que al dar de baja un contrato, éste no será eliminado del sistema completamente, sino que se mantendrá en un estado que indique que ha sido dado de baja.

Para un funcionamiento coherente del sistema, deberá por tanto existir la posibilidad de gestionar las empresas, para mantener sus datos en el sistema y determinarlas en el contrato.

BLOQUE DE REGISTROS DE CONSUMO

A partir de la información suministrada por las empresas contratadas, anteriormente en formato XML (Endesa), actualmente en formato Excel (Unión Fenosa) o cualquier otro formato digital, se deben poder insertar en el sistema las lecturas de consumos de energía que se hayan generado.

Sin embargo, Unión Fenosa no nos proporciona información sobre los cuartos horarios de consumo que se realizan, por ello este bloque sólo tiene sentido para facturas de Endesa que sí los proporcionaban.

Para hacer esto ya se ha desarrollado [1] una herramienta adecuada, por lo que la interfaz deberá enlazar dicha herramienta y proporcionar al usuario un modo sencillo de introducir los ficheros de datos.

Deberá conseguirse la mayor flexibilidad posible, ya que actualmente aún no se ha estipulado un modelo de presentación de la factura y cada compañía podría proporcionar los datos en formatos diferentes.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

BLOQUE DE FACTURACIÓN

Al igual que los registros de consumo mencionados en el apartado anterior, se deberá introducir el resto de los datos que aparecerán en una factura, que vendrá determinada por el contrato al que pertenece y a través de éste el CUPS. Ya existe, como en el caso anterior, una herramienta desarrollada con este propósito para la inserción de las facturas de Endesa, por lo que la interfaz deberá efectuar como puente entre el usuario y el programa.

Se ha desarrollado también una herramienta que se adapte al formato de las facturas de Unión Fenosa, que a través de un sencillo interfaz permita la ejecución de este.

En cualquier caso, se pide la posible inserción de una factura por parte de un usuario, en previsión de datos diferentes o errores en el formato enviado por la compañía suministradora.

MÓDULO DE CONSULTA

Módulo encargado de proporcionar al usuario la información almacenada en la Base de Datos de una manera amigable y con distintas capacidades de filtrado. Todos los usuarios tendrán acceso a las consultas, aunque no a todos los datos contenidos en ellas, sino que sólo a aquellos datos que estén relacionados con los CUPS a los que tiene acceso.

Como ya se ha comentado, el acceso a los CUPS se realiza mediante una agrupación lógica denominada Rol, que podrá tener permisos para efectuar unas u otras acciones.

BLOQUE DE FILTRADO

Realizará consultas para mostrar unos listados convenientemente formateados de los elementos disponibles. Estas consultas serán de ayuda a la hora de insertar nuevos datos o realizar búsquedas.

Como su propio nombre indica, las búsquedas deberán estar filtradas por los campos correspondientes a cada objeto que se refieran.

BLOQUE DE CÁLCULO

En este bloque se ejecutarán las consultas que resultan más complejas que el simple filtrado de datos. Se refieren en su mayor medida a cálculos y consultas relacionadas con los datos obtenidos en los registros de consumo y la facturación.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Más al detalle, el módulo se encargará de la generación de los informes solicitados, que deberán ser exportados a diferentes formatos, como por ejemplo Microsoft Excel.

Los informes a generar son los siguientes:

- Reproducir la factura comprobando si es correcta. También se encargará de generar las alarmas necesarias en caso de que no lo sea.
- Mostrar una tabla con las facturas procesadas.
- Consumos por CUPS.
- Consumo de energía activa total y por periodos, ordenada por área, centros, tarifa, potencia contratada y CUPS.
- Potencia activa máxima, total y por periodos, ordenada por área, centros, tarifa, potencia contratada y CUPS.
- Gasto total ordenado por área, centros, fecha, tarifa y CUPS.
- Gasto debido a excesos de potencia, informe ordenado por área, centro, tarifa y CUPS.
- Gasto debido a excesos de reactiva, informe ordenado por área, centro, tarifa, potencia y CUPS.
- Gasto en concepto de alquiler de equipos o gestión de medida, ordenada por área, centros, tarifa, potencia contratada y CUPS.
- Calcular la producción de CO_2.
- Informe de los suministros de socorro existentes.
- Informe de los solapamientos o huecos en las facturas.

Se presentarán además gráficos de barras para los datos individuales del mes y de líneas con marcadores para los datos acumulados.

También se podrá tener acceso a históricos y acumulados por distintos periodos o centros y áreas de los importes y consumos.

Todos estos informes deberán poder ser exportados a diferentes formatos digitales, como Microsoft Excel o PDF además de dar la posibilidad de imprimirlos.

Hay que tener en cuenta que los informes son generados mediante procedimientos almacenados en la propia base de datos, con lo que el sistema deberá preocuparse únicamente de mostrarlos al usuario de una manera cómoda y amigable.

Algunos de los informes sólo tienen sentido para las facturas de Endesa, ya que Unión Fenosa no nos proporciona como ya se ha mencionado antes los cuartos horarios de consumo y algún cálculo necesario no se podrá realizar.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

BLOQUE DE SIMULACIÓN

Se deben poder hacer simulaciones, sin repercusión sobre los datos reales, sobre tarifas o consumos ficticios creándose a partir de tarifas o consumos nuevos o ya existentes, para obtener datos sobre posibles mejoras en el precio con esas nuevas opciones, teniendo también en cuenta los consumos reales.

Además, en los informes generados a partir de las simulaciones, deberá indicarse de manera clara que se trata de simulaciones y no de datos reales.

Concretamente, las simulaciones que se realizarán (ya desarrolladas en la base de datos al igual que sucede con los informes) son las siguientes:

- Cambio de potencia contratada.
- Cambio de tarifa contratada.

Aunque este bloque sólo tendrá sentido en la facturación de Endesa por las mismas razones dadas en el bloque de cálculo.

ALARMAS

En las reuniones mantenidas con el personal del SNS-O, se decidió implementar también una funcionalidad de alarmas. Sin embargo, en nuevas reuniones no se le dada mucha importancia. Las alarmas podrán ser definidas por un usuario y corresponderán a los CUPS a los que tenga acceso mediante los Roles definidos.

Las alarmas a considerar son las siguientes:

- Alarma si en una fecha determinada no se han insertado las facturas necesarias.
- Alarma si al introducir los consumos y comprobar la factura se encuentran datos incorrectos, en la facturación de Endesa.
- Alarmas de umbrales de activa y reactiva.

Las alarmas generarán una notificación por email al encargado de revisarla, así como un mensaje de alerta al iniciar la aplicación, si esta no ha sido revisada. Las alarmas se podrán consultar para darles solución.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

2.1.2 ANÁLISIS DE CASOS DE USO

Siguiendo la metodología de Proceso Unificado (UP/RUP) los requisitos especificados en el apartado anterior deben ser analizados y convertidos en Casos de Uso.

Los actores que se han detallado son los usuarios, divididos en Administrador, Usuario de Datos y Usuario de Consulta.

En todos los Casos de Uso existe la posibilidad de que se produzca un error en la Base de Datos. Omitimos esa situación en los diagramas para simplificar, teniendo en cuenta que si eso sucede se mostrará un error al usuario.

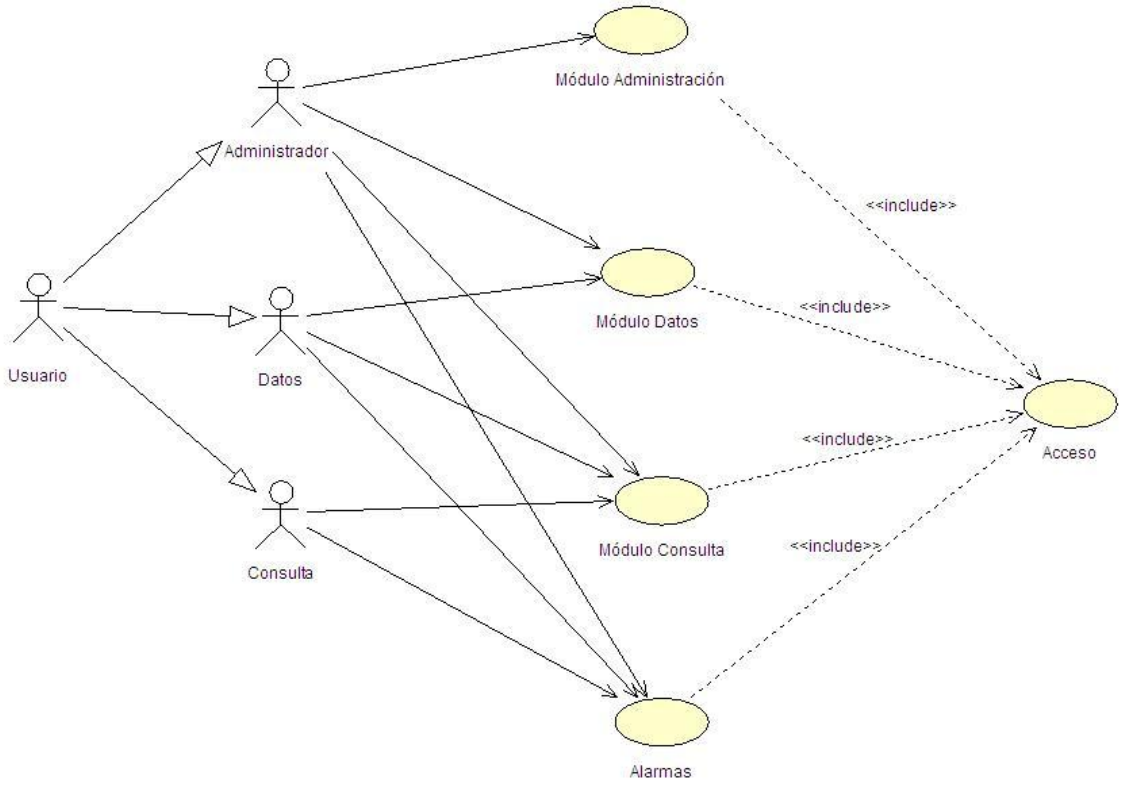


Figura 3: Diagrama de Casos de Uso del Sistema

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASOS DE USO: ACCESO

Descripción: Acceso al sistema
Precondición: Ninguna
Camino básico: 1. El usuario manda su usuario y contraseña. 2. El sistema busca en la base de datos. 3. La base de datos devuelve los roles del usuario. 4. El usuario recibe los roles disponibles. 5. El usuario elige un rol. 6. El sistema le da acceso con ese rol.
Camino alternativo: 7. En 2, no existen los datos del usuario. 8. La base de datos devuelve el error. 9. El sistema comunica el error al usuario.
Postcondición: El usuario ha accedido al sistema con un Rol determinado.

Tabla 1: Caso de Uso Acceso al Sistema

MÓDULO DE ADMINISTRACIÓN

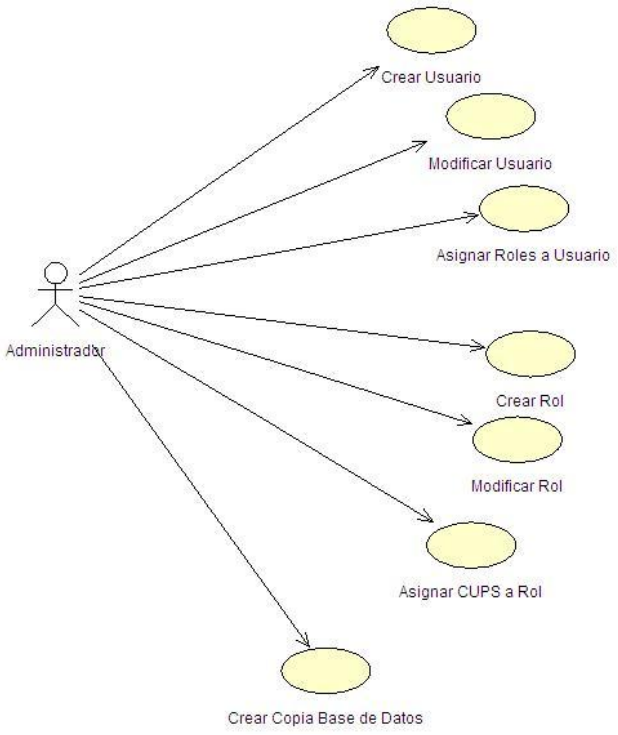


Figura 4: Diagrama de Casos de Uso, Módulo de Administración

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: CREAR USUARIO

Descripción: Creación de un nuevo usuario en el sistema
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce los datos del nuevo usuario. 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al administrador.
Postcondición: El usuario se ha creado correctamente.

Tabla 2: Caso de Uso Crear Usuario

CASO DE USO: MODIFICAR USUARIO

Descripción: Modificar los datos de un usuario
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador del usuario. 2. El sistema pide a la BD los datos del usuario. 3. El sistema devuelve los datos del usuario. 4. El administrador introduce los nuevos datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 3, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: El usuario se ha modificado correctamente.

Tabla 3: Caso de Uso Modificar Usuario

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: CREAR ROL

Descripción: Creación de un rol de acceso al sistema
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce los datos del nuevo rol. 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al administrador.
Postcondición: El Rol se ha creado correctamente.

Tabla 4: Caso de Uso Crear Rol

CASO DE USO: MODIFICAR ROL

Descripción: Modificar los datos de un rol
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador del rol. 2. El sistema pide a la BD los datos del usuario. 3. El sistema devuelve los datos del rol. 4. El administrador introduce los nuevos datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 3, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: El Rol se ha modificado correctamente.

Tabla 5: Caso de Uso Modificar Rol

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: ASIGNAR ROLES A USUARIO

Descripción: Asignar un conjunto de Roles a un usuario
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El sistema obtiene de la base de datos los Roles disponibles y se los muestra al administrador. 2. El administrador introduce el identificador de usuario. 3. El administrador introduce los roles que tiene que tener asociados. 4. El sistema manda los datos a la Base de Datos. 5. Se crean las asociaciones nuevas y se eliminan las antiguas.
Camino alternativo:
Postcondición: Se han asignado los nuevos Roles al Usuario.

Tabla 6: Caso de Uso Asignar Roles a Usuario

CASO DE USO: ASIGNAR CUPS A ROL

Descripción: Asignar un conjunto de CUPS a un usuario
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El sistema obtiene de la base de datos los CUPS disponibles y se los muestra al administrador. 2. El administrador introduce el identificador de Rol. 3. El administrador introduce los CUPS que tiene que tener asociados. 4. El sistema manda los datos a la Base de Datos. 5. Se crean las asociaciones nuevas y se eliminan las antiguas.
Camino alternativo:
Postcondición: Se han asignado los nuevos CUPS al Rol.

Tabla 7: Caso de Uso Asignar CUPS a Rol

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: CREAR COPIA BASE DE DATOS

Descripción: Crear una copia de la base de Datos
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador manda al sistema crear una copia de la base de datos. 2. El sistema manda los datos a la Base de Datos.
Camino alternativo:
Postcondición: Se ha creado la copia de la base de datos correctamente.

Tabla 8: Caso de Uso Crear Copia Base de Datos

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MÓDULO DE DATOS

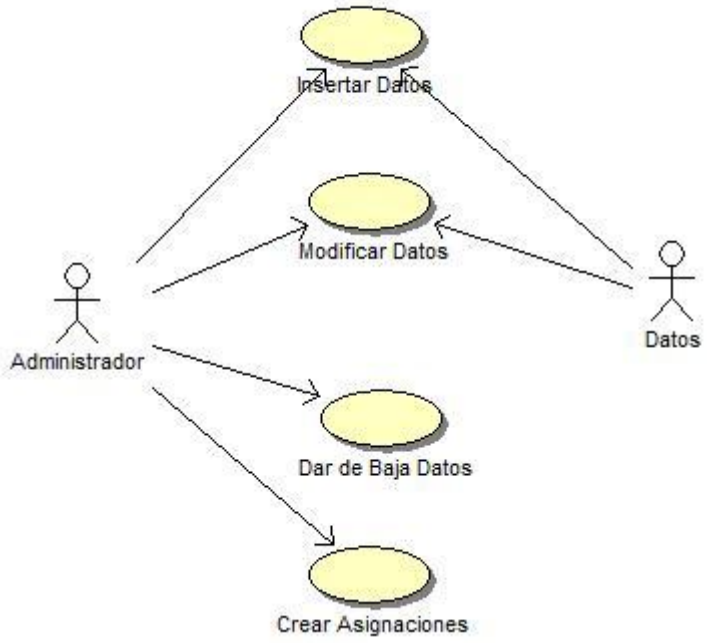


Figura 5: Diagrama de Casos de Uso, Módulo de Datos

INSERTAR DATOS

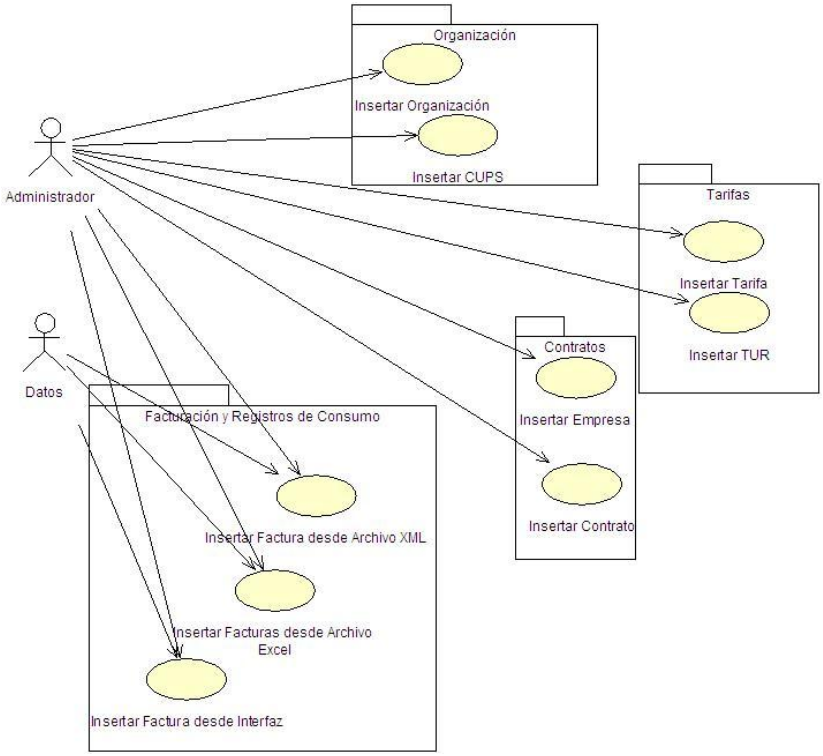


Figura 6: Diagrama de Casos de Uso, Insertar Datos

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: INSERTAR ORGANIZACIÓN

Descripción: Inserción de una organización
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce los datos de la Organización 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al administrador.
Postcondición: La Organización se ha insertado correctamente.

Tabla 9: Caso de Uso Insertar Organización

CASO DE USO: INSERTAR CUPS

Descripción: Inserción de un CUPS
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce los datos del CUPS. 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al administrador.
Postcondición: El CUPS se ha insertado correctamente.

Tabla 10: Caso de Uso Insertar CUPS

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: INSERTAR TARIFA

Descripción: Inserción de una Tarifa
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador pide o no los datos de alguna Tarifa, para insertarla a partir de ella. 2. El administrador introduce los datos de la Tarifa. 3. El administrador introduce los periodos de la Tarifa. 4. El sistema comprueba los datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 2, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: La Tarifa se ha insertado correctamente.

Tabla 11: Caso de Uso Insertar Tarifa

CASO DE USO: INSERTAR TARIFA DE ÚLTIMO RECURSO

Descripción: Inserción de una Tarifa de Último Recurso
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce los datos de la TUR. 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al administrador.
Postcondición: La TUR se ha insertado correctamente.

Tabla 12: Caso de Uso Insertar Tarifa de Último Recurso

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: INSERTAR EMPRESA

Descripción: Inserción de una Empresa
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce los datos de la Empresa. 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al administrador.
Postcondición: La Empresa se ha insertado correctamente.

Tabla 13: Caso de Uso Insertar Empresa

CASO DE USO: INSERTAR CONTRATO

Descripción: Inserción de un Contrato
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador recibe los datos de las Empresas disponibles. 2. El administrador introduce los datos del Contrato. 3. El administrador indica la Empresa. 4. El sistema comprueba los datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 3, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: El Contrato se ha insertado correctamente.

Tabla 14: Caso de Uso Insertar Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: INSERTAR FACTURA DESDE ARCHIVO XML

Descripción: Inserción de una Factura desde un archivo XML
Precondición: Estar identificado con perfil de Administrador o Datos
Camino básico: 1. El usuario envía el archivo con la factura. 2. El sistema analiza el archivo. 3. El sistema envía a la BD los datos de la factura y los consumos.
Camino alternativo: 4. En 3, el archivo no está bien construido. 5. Se muestra error al usuario. 6. Después de 4 si los datos no coinciden. 7. Se genera una alarma, además de insertarlos.
Postcondición: La Factura y los consumos se han insertado correctamente.

Tabla 15: Caso de Uso Insertar Factura desde Archivo XML

CASO DE USO: INSERTAR FACTURAS DESDE ARCHIVO EXCEL

Descripción: Inserción de Facturas desde un archivo Excel
Precondición: Estar identificado con perfil de Administrador o Datos
Camino básico: 1. El usuario envía el archivo con las facturas. 2. Si se importa directamente el sistema analiza el archivo. 3. El sistema envía a la BD los datos de la factura y los consumos. 4. Si se guarda en borrador y luego se valida. 5. El sistema envía a la BD los datos de la factura y los consumos.
Camino alternativo: 6. En 2 o en 4, hay errores de cálculo en el archivo o al insertar en la base de datos. 7. Se muestra errores al usuario y se genera un Excel con colores indicando errores.
Postcondición: Las Facturas y los consumos se han insertado correctamente.

Tabla 16: Caso de Uso Insertar Facturas desde Archivo Excel

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: INSERTAR FACTURA DESDE INTERFAZ

Descripción: Inserción de una Factura desde la interfaz
Precondición: Estar identificado con perfil de Administrador o Datos
Camino básico: 1. El usuario introduce los datos de la Factura. 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al administrador.
Postcondición: La Factura se ha insertado correctamente.

Tabla 17: Caso de Uso Insertar Factura desde Interfaz

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MODIFICAR DATOS

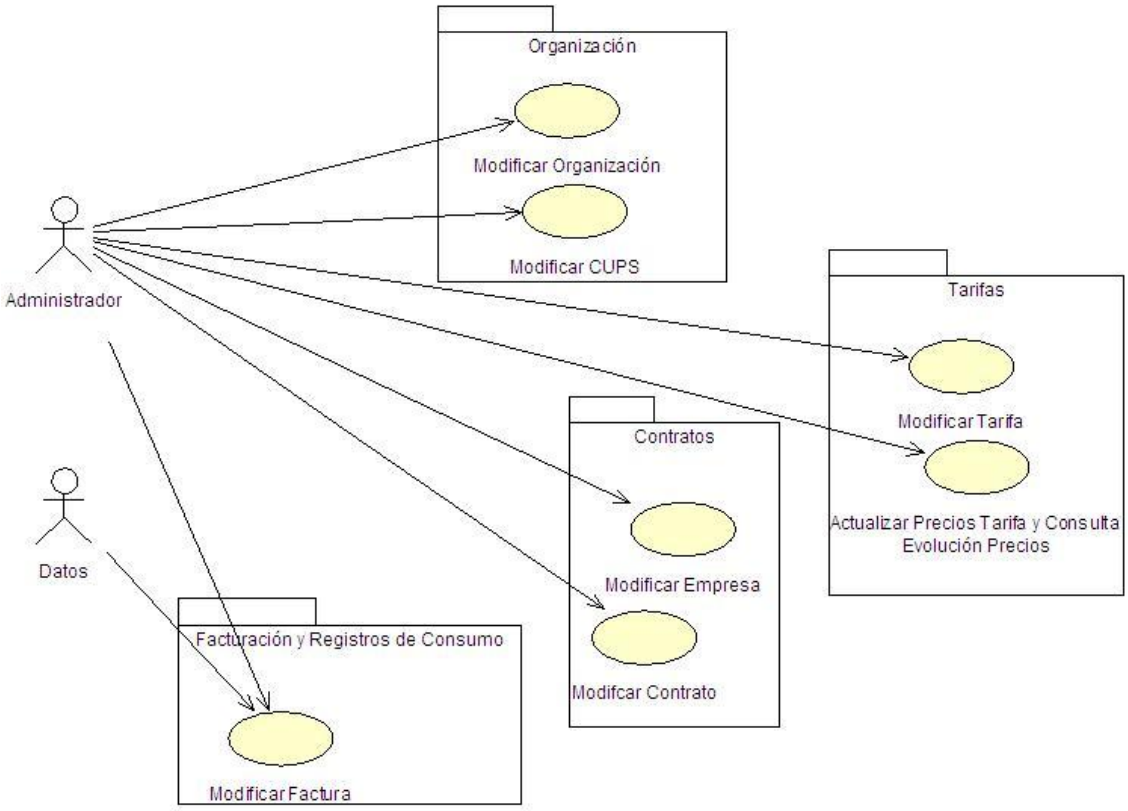


Figura 7: Diagrama de Casos de Uso, Modificar Datos

CASO DE USO: MODIFICAR ORGANICACIÓN

Descripción: Modificar los datos de una Organización
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador de la Organización. 2. El sistema pide a la BD los datos de la Organización. 3. El sistema devuelve los datos de la Organización. 4. El administrador introduce los nuevos datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 3, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: La Organización se ha modificado correctamente.

Tabla 18: Caso de Uso Modificar Organización

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: MODIFICAR CUPS

Descripción: Modificar los datos de un CUPS
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador del CUPS. 2. El sistema pide a la BD los datos del CUPS. 3. El sistema devuelve los datos del CUPS. 4. El administrador introduce los nuevos datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 3, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: El CUPS se ha modificado correctamente.

Tabla 19: Caso de Uso Modificar CUPS

CASO DE USO: MODIFICAR TARIFA

Descripción: Modificar los datos de una Tarifa
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador de la Tarifa. 2. El sistema pide a la BD los datos de la Tarifa 3. El sistema devuelve los datos de la Tarifa. 4. El usuario introduce los nuevos datos fijos de Tarifa. 5. El usuario introduce los nuevos periodos de la Tarifa. 6. El usuario introduce los nuevos datos variables de la Tarifa. 7. El sistema envía todos los datos a la BD. 8. Se cambian los datos fijos. 9. Se cambian los periodos. 10. Se insertan los nuevos datos variables.
Camino alternativo: 11. En 6, alguno de los datos es erróneo. 12. Se muestra error al administrador.
Postcondición: La Tarifa se ha modificado correctamente.

Tabla 20: Caso de Uso Modificar Tarifa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: ACTUALIZAR PRECIOS TARIFA Y CONSULTA EVOLUCIÓN PRECIOS

Descripción: Actualizar los precios la activa y potencia de una Tarifa
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador de la Tarifa. 2. El sistema pide a la BD los precios de la activa y de la potencia y datos de la Tarifa. 3. El sistema devuelve los precios y datos de la Tarifa y permite visualizar la evolución de los precios de la Tarifa con el tiempo. 4. Si el usuario introduce los nuevos precios de la activa y la potencia y la nueva fecha de inicio de la Tarifa. 5. El sistema envía todos los datos a la BD. 6. Se cambian los precios y la fecha. 7. Si el usuario quiere visualizar la evolución de los precios de la tarifa en el tiempo 8. El sistema genera una gráfica con los precios
Camino alternativo: 9. En 4, alguno de los datos es erróneo. 10. Se muestra error al administrador.
Postcondición: La precios de la Tarifa se han modificado correctamente o se ha generado correctamente la gráfica.

Tabla 21: Caso de Uso Actualizar Precio Tarifa y Consulta Evolución Precios

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: MODIFICAR EMPRESA

Descripción: Modificar los datos de una Empresa
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador de la Empresa. 2. El sistema pide a la BD los datos de la Empresa. 3. El sistema devuelve los datos de la Empresa. 4. El administrador introduce los nuevos datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 3, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: La Empresa se ha modificado correctamente.

Tabla 22: Caso de Uso Modificar Empresa

CASO DE USO: MODIFICAR CONTRATO

Descripción: Modificar los datos de un Contrato
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador recibe los datos de las Empresas disponibles. 2. El administrador introduce el identificador del Contrato. 3. El sistema pide a la BD los datos del Contrato. 4. El sistema devuelve los datos del Contrato. 5. El administrador introduce los nuevos datos. 6. El sistema manda los datos a la Base de Datos.
Camino alternativo: 7. En 4, los datos son erróneos. 8. Se muestra error al administrador.
Postcondición: El contrato se ha modificado correctamente.

Tabla 23: Caso de Uso Modificar Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: MODIFICAR FACTURA

Descripción: Modificar los datos de una Factura
Precondición: Estar identificado con perfil de Administrador o Datos
Camino básico: 1. El administrador introduce el identificador de la Factura. 2. El sistema pide a la BD los datos de la Factura. 3. El sistema devuelve los datos de la Factura. 4. El administrador introduce los nuevos datos. 5. El sistema manda los datos a la Base de Datos.
Camino alternativo: 6. En 3, los datos son erróneos. 7. Se muestra error al administrador.
Postcondición: La Factura se ha modificado correctamente.

Tabla 24: Caso de Uso Modificar Factura

DAR DE BAJA DATOS

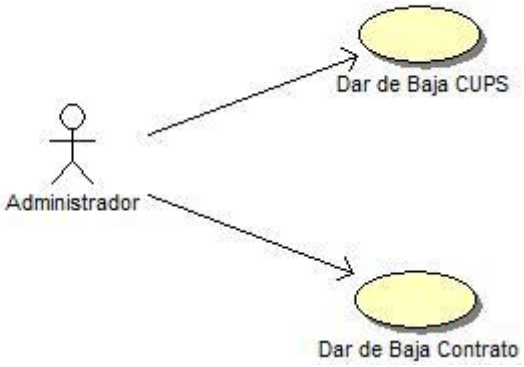


Figura 8: Diagrama de Casos de Uso, Dar de Baja Datos

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: DAR DE BAJA CUPS

Descripción: Indicar que un CUPS ya no está activo
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador del CUPS. 2. El administrador introduce la fecha de baja. 3. El sistema envía el identificador y la fecha de baja del CUPS a la BD.
Camino alternativo: 4. En 2, la fecha es errónea. 5. Se muestra error al administrador.
Postcondición: El CUPS ha sido dado de Baja.

Tabla 25: Caso de Uso Dar de Baja CUPS

CASO DE USO: DAR DE BAJA CONTRATO

Descripción: Indicar que un Contrato ha terminado
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El administrador introduce el identificador del Contrato. 2. El administrador introduce la fecha de baja. 3. El sistema envía el identificador y la fecha de baja del Contrato a la BD.
Camino alternativo: 4. En 2, la fecha es errónea. 5. Se muestra error al administrador.
Postcondición: El Contrato ha sido dado de Baja.

Tabla 26: Caso de Uso Dar de Baja Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CREAR ASIGNACIONES

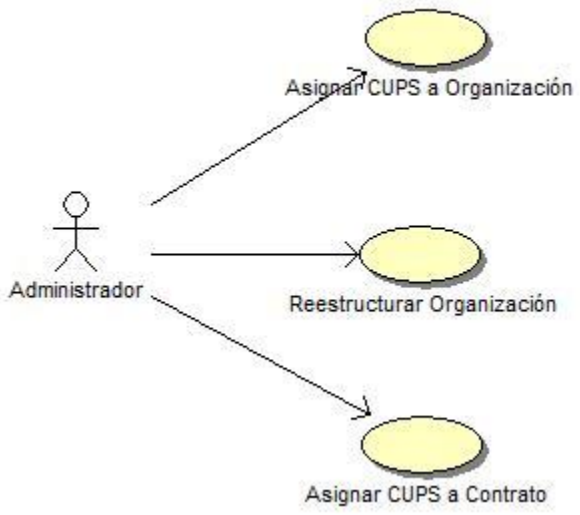


Figura 9: Diagrama de Casos de Uso, Crear Asignaciones

CASO DE USO: ASOCIAR CUPS A ORGANIZACIÓN

Descripción: Asignar un conjunto de CUPS a una Organización
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El sistema obtiene de la base de datos los CUPS disponibles y se los muestra al administrador. 2. El administrador introduce el identificador de la Organización. 3. El administrador introduce los CUPS que tiene que tener asociados. 4. El sistema manda los datos a la Base de Datos. 5. Se crean las asociaciones nuevas y se eliminan las antiguas.
Camino alternativo:
Postcondición: Se han asignado los nuevos CUPS a la Organización.

Tabla 27: Caso de Uso Asociar CUPS a Organización

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: REESTRUCTURAR ORGANIZACIÓN

Descripción: Cambiar la estructura de una organización
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El sistema obtiene de la base de datos las Organizaciones disponibles. 2. El administrador introduce el identificador de la Organización. 3. El administrador introduce las Organizaciones inferiores que tienen que tener asociadas. 4. El sistema manda los datos a la Base de Datos. 5. Se crean las asociaciones nuevas y se eliminan las antiguas.
Camino alternativo: 6. En 3, la asociación no es válida 7. Se muestra error al administrador.
Postcondición: Se han asignado las nuevas organizaciones inferiores a la organización.

Tabla 28: Caso de Uso Reestructurar Organización

CASO DE USO: ASIGNAR CUPS A CONTRATO

Descripción: Asignar un único CUPS a un Contrato
Precondición: Estar identificado con perfil de Administrador
Camino básico: 1. El sistema obtiene de la base de datos los CUPS disponibles y se los muestra al administrador. 2. El administrador introduce el identificador del Contrato. 3. El administrador introduce los CUPS que tiene que tener asociados. 4. El sistema manda los datos a la Base de Datos. 5. Se crean las asociaciones nuevas y se eliminan las antiguas.
Camino alternativo:
Postcondición: Se ha asignado el CUPS al Contrato.

Tabla 29: Caso de Uso Asignar CUPS a Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MÓDULO DE CONSULTA

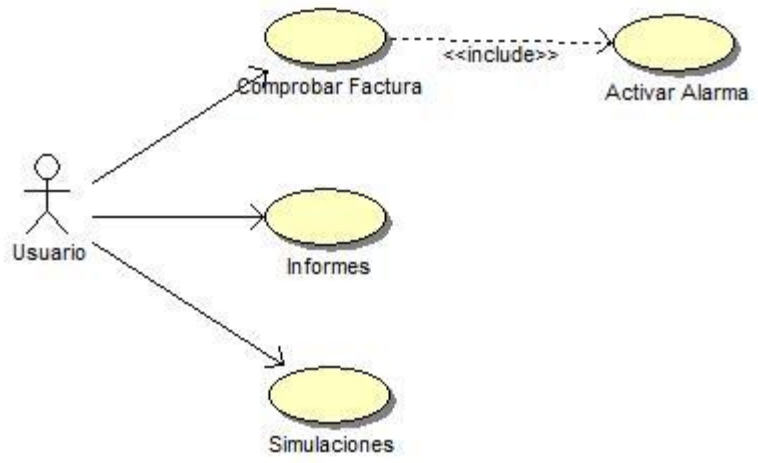


Figura 10: Diagrama de Casos de Uso, Módulo de Consulta

CASO DE USO: COMPROBAR FACTURA

Descripción: Comprobar si los datos de una Factura se corresponden con los Registros de Consumo
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario introduce la factura. 2. El sistema comprueba la factura. 3. Si los datos son correctos la factura pasa a estar validada.
Camino alternativo: 4. En 3, si los datos son incorrectos se muestra el error al usuario. 5. Se generan las alarmas (caso de uso Activar Alarma).
Postcondición: Factura comprobada y si es necesario alarmas activadas.

Tabla 30: Caso de Uso Comprobar Factura

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: ACTIVAR ALARMA

Descripción: Activación de una alarma
Precondición: Suceso extraño en el sistema.
Camino básico: 1. El sistema analiza el suceso. 2. Se genera la Alarma. 3. Se introduce la Alarma en la BD. 4. Se envía correo electrónico al usuario responsable.
Camino alternativo:
Postcondición: Se ha creado una Alarma.

Tabla 31: Caso de Uso Activar Alarma

INFORMES

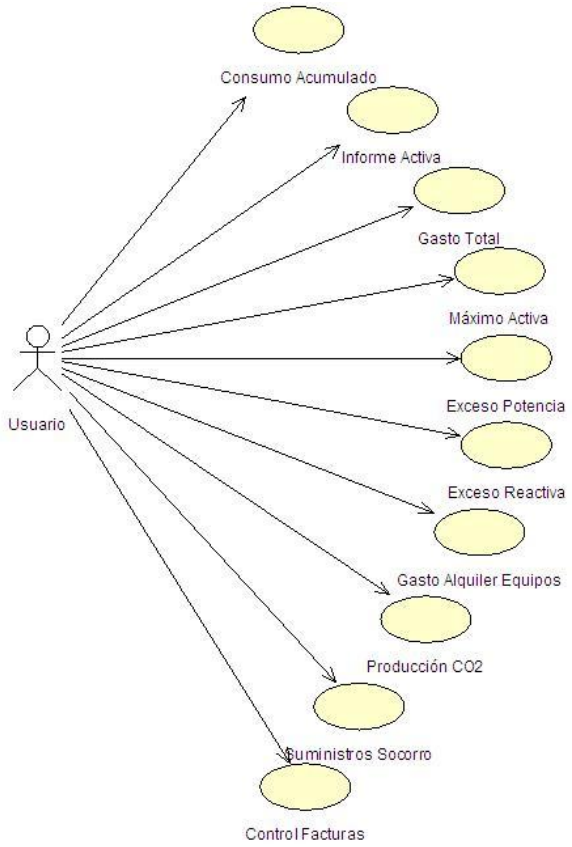


Figura 11: Diagrama de Casos de Uso, Informes

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: CONSUMO ACUMULADO

Descripción: Generación del Informe de Consumo Acumulado
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Consumo Acumulado. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 32: Caso de Uso Consumo Acumulado

CASO DE USO: INFORME ACTIVA

Descripción: Generación del Informe de Energía Activa
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Energía Activa. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 33: Caso de Uso Informe Activa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: GASTO TOTAL

Descripción: Generación del Informe de Gasto Total
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Gasto Total. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 34: Caso de Uso Gasto Total

CASO DE USO: EXCESO REACTIVA

Descripción: Generación del Informe de Exceso de Energía Reactiva
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Exceso de Energía Reactiva. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 35: Caso de Uso Exceso Reactiva

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: MÁXIMO ACTIVA

Descripción: Generación del Informe de Máximo consumo de Energía Activa
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Máximo consumo de Energía Activa. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 36: Caso de Uso Máximo Activa

CASO DE USO: EXCESO POTENCIA

Descripción: Generación del Informe de Excesos de Potencia
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Excesos de Potencia. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 37: Caso de Uso Exceso Potencia

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: GASTO ALQUILER DE EQUIPOS

Descripción: Generación del Informe de Gasto por Alquiler de Equipos
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Gasto por Alquiler de Equipos. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 38: Caso de Uso Gasto Alquiler de Equipos

CASO DE USO: PRODUCCIÓN CO_2

Descripción: Generación del Informe de Producción de CO_2
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Producción de CO_2. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 39: Caso de Uso Producción de CO2

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CASO DE USO: SUMINISTROS SOCORRO

Descripción: Generación del Informe de Suministros de Socorro
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Suministros de Socorro. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 40: Caso de Uso Suministros Socorro

CASO DE USO: CONTROL FACTURAS

Descripción: Generación del Informe de Control de Facturas
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario hace la petición del Informe 2. El sistema llama al procedimiento almacenado que generará el informe de Control de Facturas. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 41: Caso de Uso Control Facturas

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

SIMULACIONES

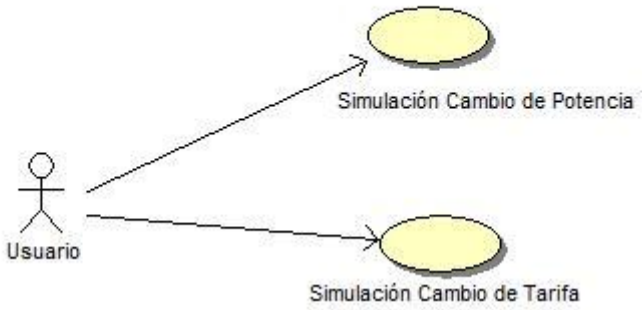


Figura 12: Diagrama de Casos de Uso, Simulaciones

SIMULACIÓN CAMBIO DE POTENCIA

Descripción: Simulación de gasto tras un cambio de potencia
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario introduce los datos de la simulación. 2. El sistema llama al procedimiento almacenado que generará el informe de la simulación de Cambio de Potencia. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 42: Caso de Uso Cambio de Potencia

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

SIMULACIÓN CAMBIO DE TARIFA

Descripción: Simulación de gasto tras un cambio de tarifa
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario introduce los datos de la simulación. 2. El sistema llama al procedimiento almacenado que generará el informe de la simulación de Cambio de Tarifa. 3. La BD devuelve los datos para el informe. 4. El sistema presenta los datos y dibuja los gráficos.
Camino alternativo:
Postcondición: El usuario obtiene el informe.

Tabla 43: Caso de Uso Cambio de Tarifa

ALARMAS

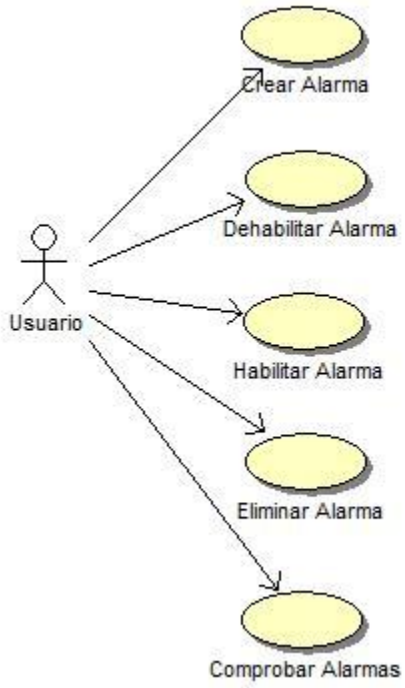


Figura 13: Diagrama de Casos de Uso, Alarmas

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CREAR ALARMA

Descripción: Creación de una nueva alarma
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario introduce los parámetros de la Alarma. 2. El sistema comprueba los datos. 3. El sistema manda los datos a la Base de Datos.
Camino alternativo: 4. En 2, los datos son erróneos. 5. Se muestra error al usuario.
Postcondición: La alarma se ha creado correctamente.

Tabla 44: Caso de Uso Crear Alarma

DESHABILITAR ALARMA

Descripción: Dejar una alarma deshabilitada, evitando que se pueda activar
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario indica el identificador de la alarma a deshabilitar. 2. El sistema envía los datos a la BD. 3. La alarma se deshabilita.
Camino alternativo:
Postcondición: La alarma se ha deshabilitado y no se podrá activar.

Tabla 45: Caso de Uso Deshabilitar Alarma

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

HABILITAR ALARMA

Descripción: Volver a habilitar una alarma deshabilitada
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario indica el identificador de la alarma a habilitar. 2. El sistema envía los datos a la BD. 3. La alarma se habilita.
Camino alternativo:
Postcondición: La alarma se ha vuelto a habilitar y podrá ser activada.

Tabla 46: Caso de Uso Habilitar Alarma

ELIMINAR ALARMA

Descripción: Eliminar una Alarma del sistema
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario indica el identificador de la alarma a eliminar. 2. El sistema envía los datos a la BD. 3. La alarma se elimina.
Camino alternativo:
Postcondición: La alarma se ha eliminado.

Tabla 47: Caso de Uso Eliminar Alarma

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

COMPROBAR ALARMAS

Descripción: Comprobar el estado de las alarmas de un usuario
Precondición: El usuario se ha identificado en el sistema
Camino básico: 1. El usuario pide consultar sus alarmas. 2. El sistema pide la información a la BD. 3. La BD envía las alarmas del usuario. 4. El sistema muestra las alarmas al usuario.
Camino alternativo:
Postcondición: El usuario obtiene un listado de sus alarmas.

Tabla 48: Caso de Uso Comprobar Alarmas

2.1.3 REQUISITOS NO FUNCIONALES

Además de los requisitos en cuanto a las funcionalidades que deberá tener la aplicación se refiere, existen otras consideraciones a tener en cuenta a la hora de diseñar e implementar el sistema:

- El sistema deberá ser capaz de manejar un gran volumen de información en lo concerniente a los consumos (en la facturación de Endesa), ya que en muchos casos de cada punto de consumo se obtiene una medida cada cuarto de hora. Este hecho no debería afectar al rendimiento del sistema.

- El sistema podría ser usado por gente poco habituada al trabajo con un ordenador, por lo que la interfaz debería ser lo más amigable posible, dando todas las opciones y facilidades para hacer más cómodo su uso. Se considera conveniente por tanto que las diferentes páginas o pantallas que se muestren en el sistema tengan una estructura y navegación homogéneas, de fácil comprensión y que las acciones que se efectúen en cada una de ellas sean lo más intuitivas posible.

- El SNS-O tiene una página web integrada en el sistema informático del Gobierno de Navarra. Para mantener la homogeneidad en su sistema, existen unas hojas de estilo y consideraciones a tener en cuenta a la hora de generar una nueva aplicación que se introducirá en él. Estas normas son de obligado cumplimiento, por lo que este sistema tendrá limitadas las opciones en cuanto al diseño gráfico.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- También las plataformas a utilizar están limitadas por el Gobierno de Navarra. Se deberán utilizar herramientas accesibles desde las plataformas de Microsoft, que son las plataformas actualmente en uso por los funcionarios navarros.

- Se deberá optimizar el aspecto gráfico de la aplicación para su visualización en monitores con una resolución de 1280 x 1024, ya que es la que utilizan los ordenadores institucionalizados en todo el estado por el ministerio de sanidad.

- Como ya se ha indicado varias veces, el Servicio Navarro de Salud – Osasunbidea es un organismo en constante cambio. Además, las empresas suministradoras de la energía no siguen el mismo patrón en cuanto al formato de las facturas. Es por esto que es imprescindible que el sistema tenga la mayor flexibilidad y adaptabilidad posible.

2.1.4 REQUISITOS HARDWARE

No existen necesidades específicas de Hardware, ya que en la actualidad el SNS-O ya posee los servidores necesarios, tanto para alojar la Base de Datos como servidores web. Lo que sí es importante, volviendo al punto anterior, es la compatibilidad con las plataformas existentes e institucionalizadas.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DISEÑO

Capítulo de diseño del sistema, en el que se diseña cada caso de uso analizado en el apartado anterior. También se presenta un diagrama de clases y la estructura general

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

3 DISEÑO

3.1 DISEÑO DE LOS CASOS DE USO

Se analizará cada caso de uso, preparando el diseño para su implementación mediante Diagramas de Secuencia. Los diagramas de secuencia comienzan cuando el usuario accede a la página correspondiente al caso de uso.

ACCESO

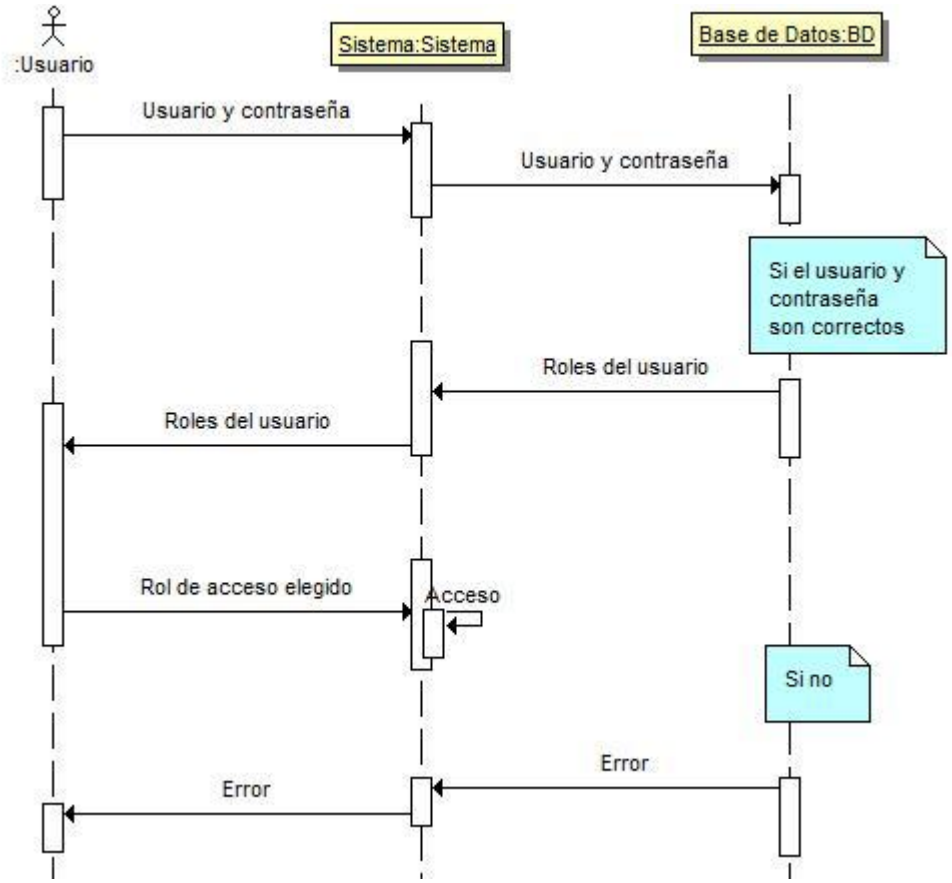


Figura 14: Diagrama de Secuencia, Acceso

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CREAR USUARIO

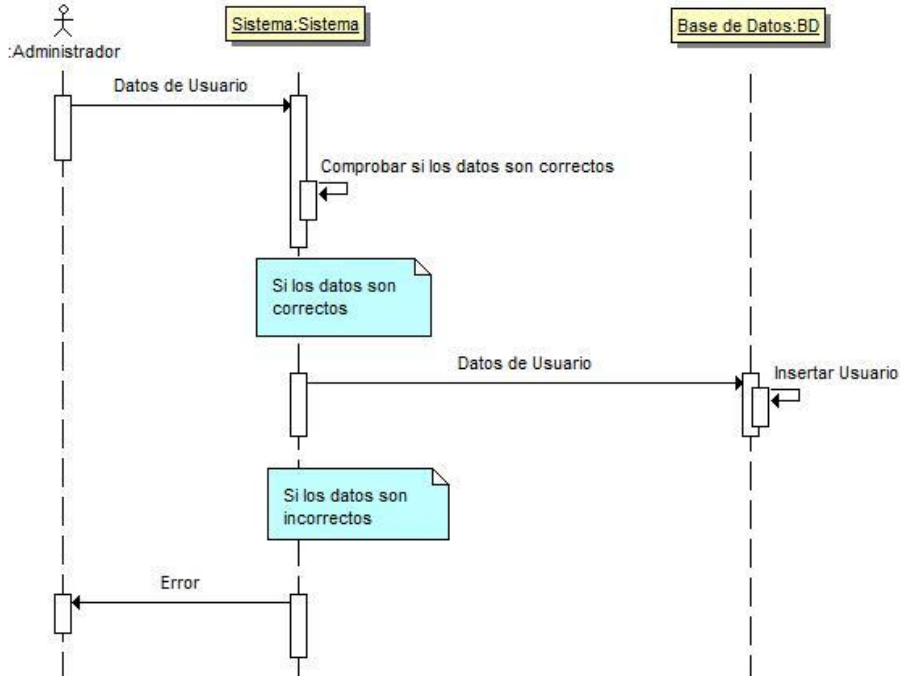


Figura 15: Diagrama de Secuencia, Crear Usuario

MODIFICAR USUARIO

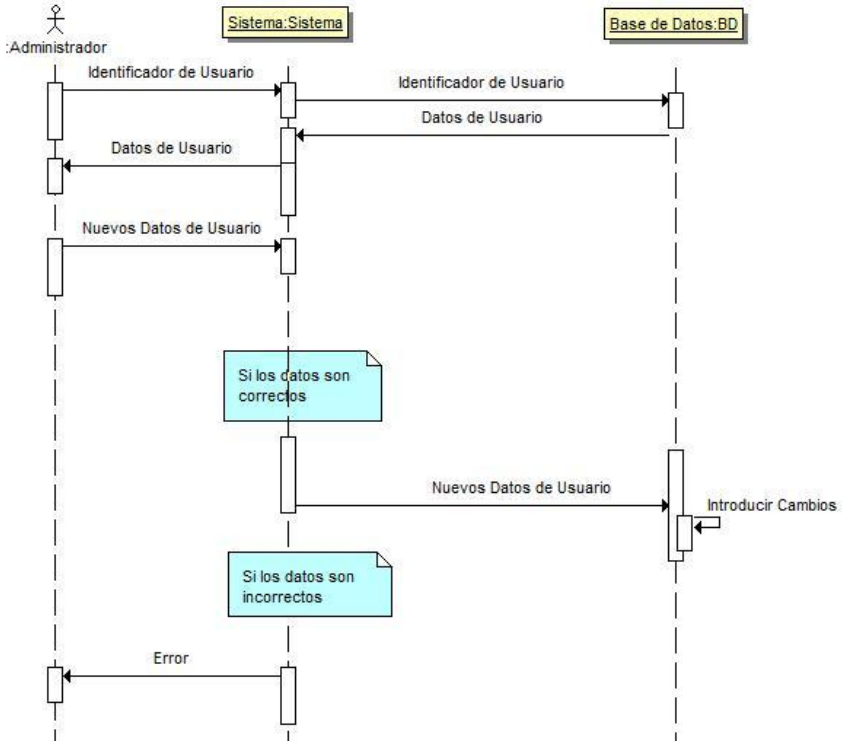


Figura 16: Diagrama de Secuencia, Modificar Usuario

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ASIGNAR ROLES A USUARIO

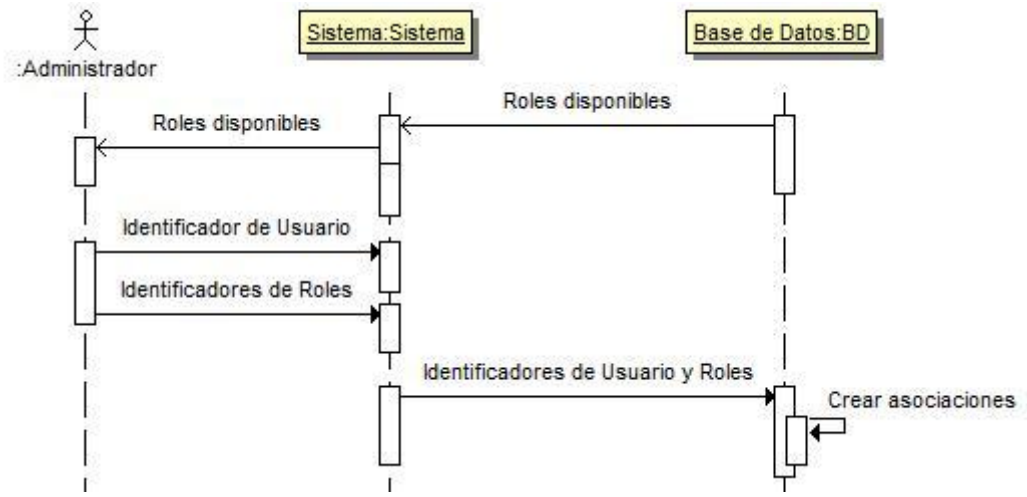


Figura 17: Diagrama de Secuencia, Asignar Roles a Usuario

CREAR ROL

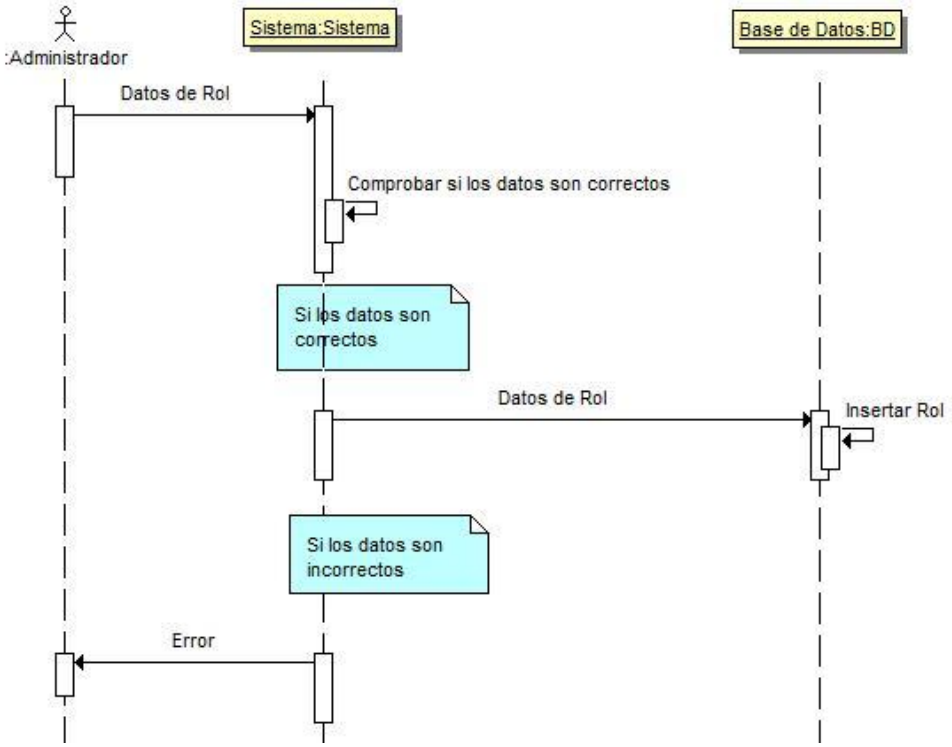


Figura 18: Diagrama de Secuencia, Crear Rol

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MODIFICAR ROL

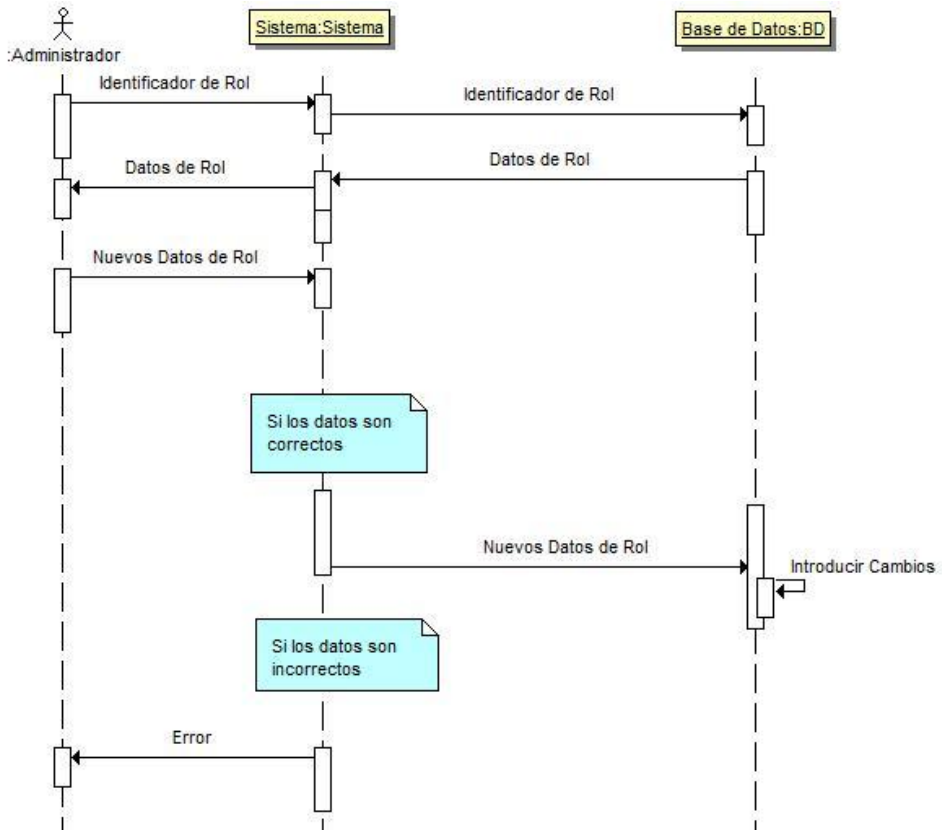


Figura 19: Diagrama de Secuencia, Modificar Rol

ASIGNAR CUPS A ROL

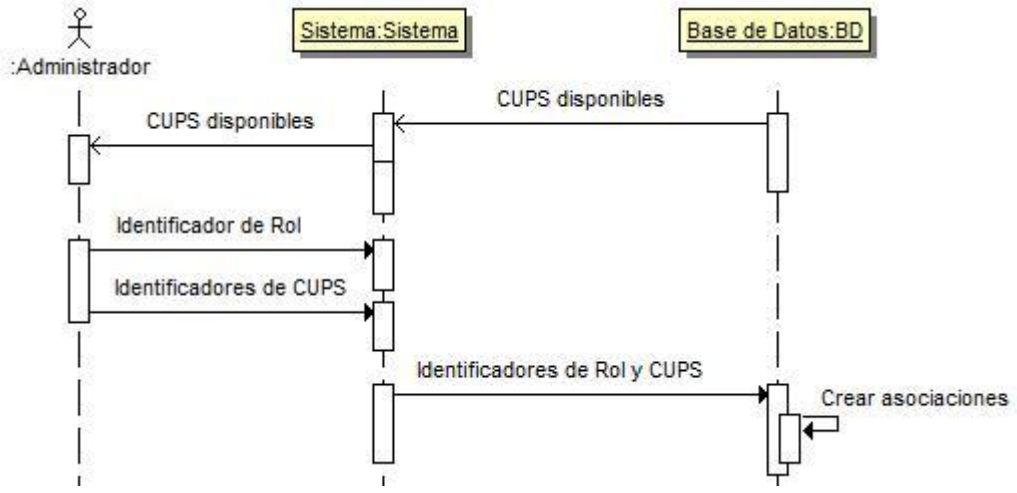


Figura 20: Diagrama de Secuencia, Asignar CUPS a Rol

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INSERTAR ORGANIZACIÓN

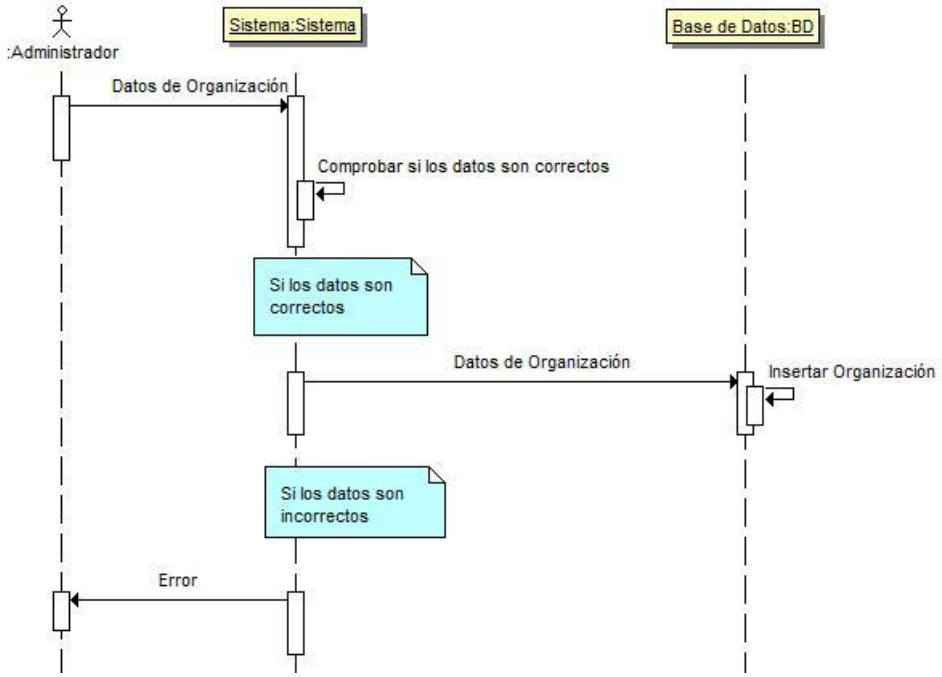


Figura 21: Diagrama de Secuencia, Insertar Organización

INSERTAR CUPS

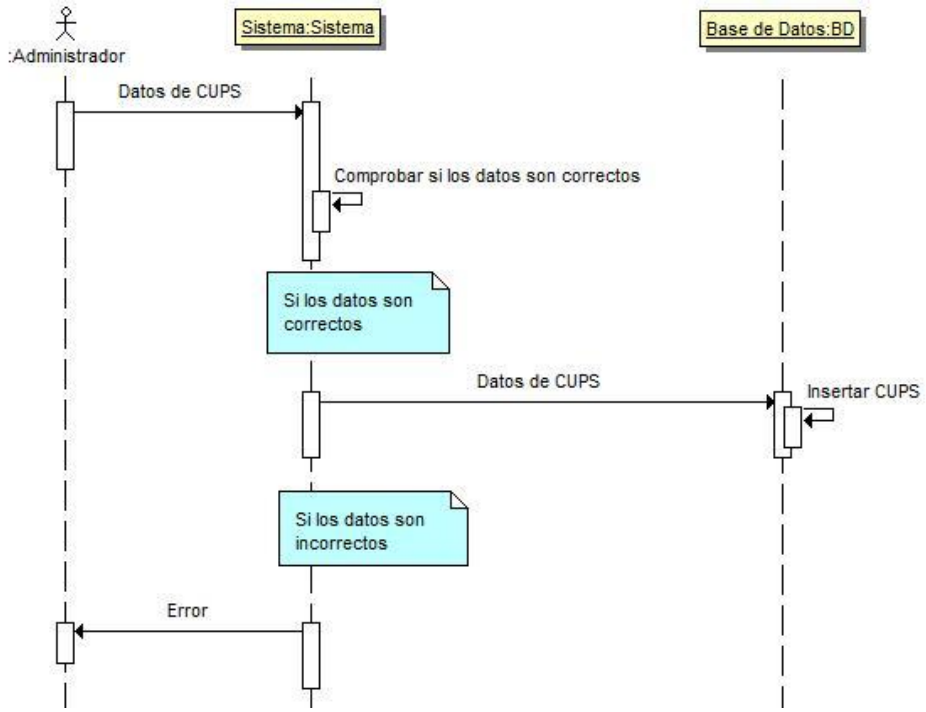


Figura 22: Diagrama de Secuencia, Insertar CUPS

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INSERTAR TARIFA

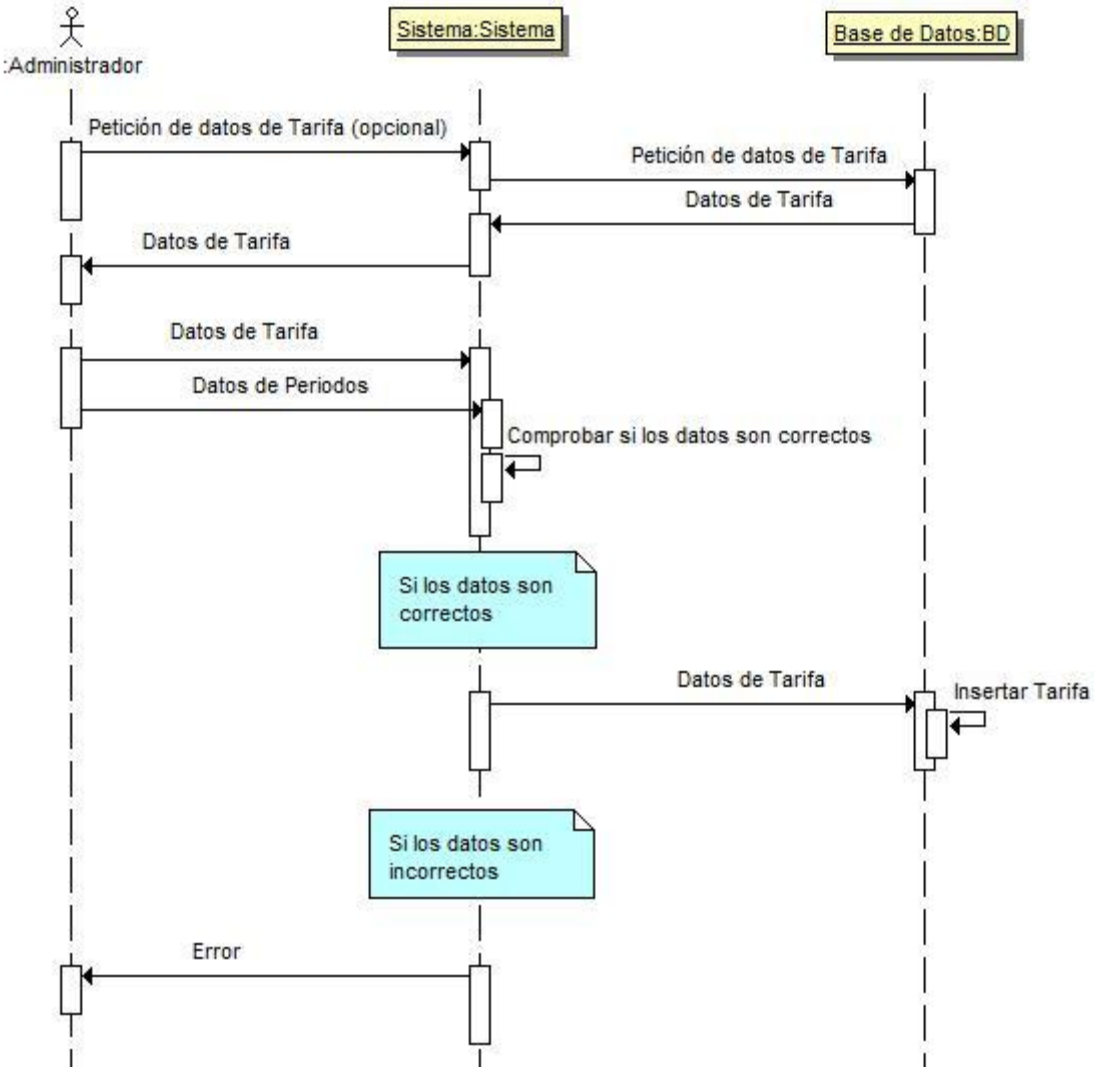


Figura 23: Diagrama de Secuencia, Insertar Tarifa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INSERTAR TARIFA DE ÚLTIMO RECURSO

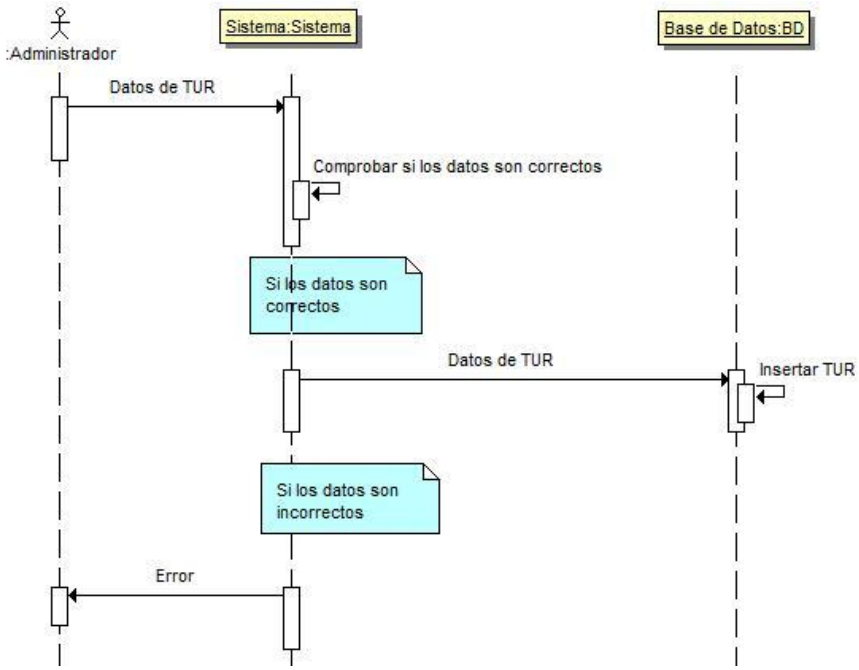


Figura 24: Diagrama de Secuencia, Insertar Tarifa de Último Recurso

INSERTAR EMPRESA

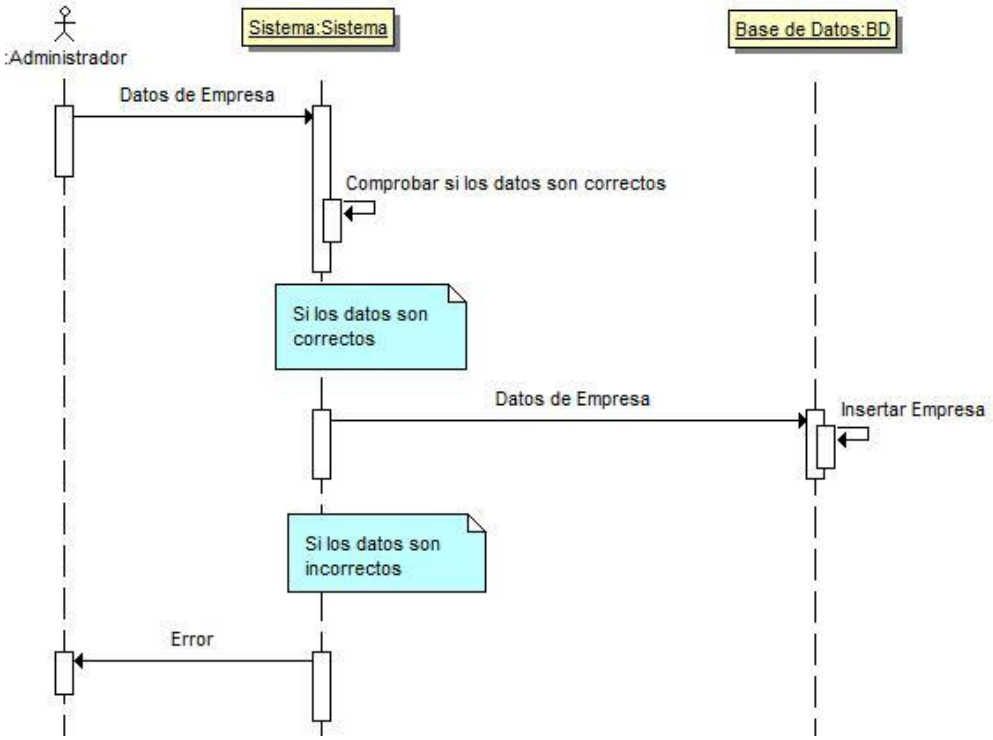


Figura 25: Diagrama de Secuencia, Insertar Empresa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INSERTAR CONTRATO

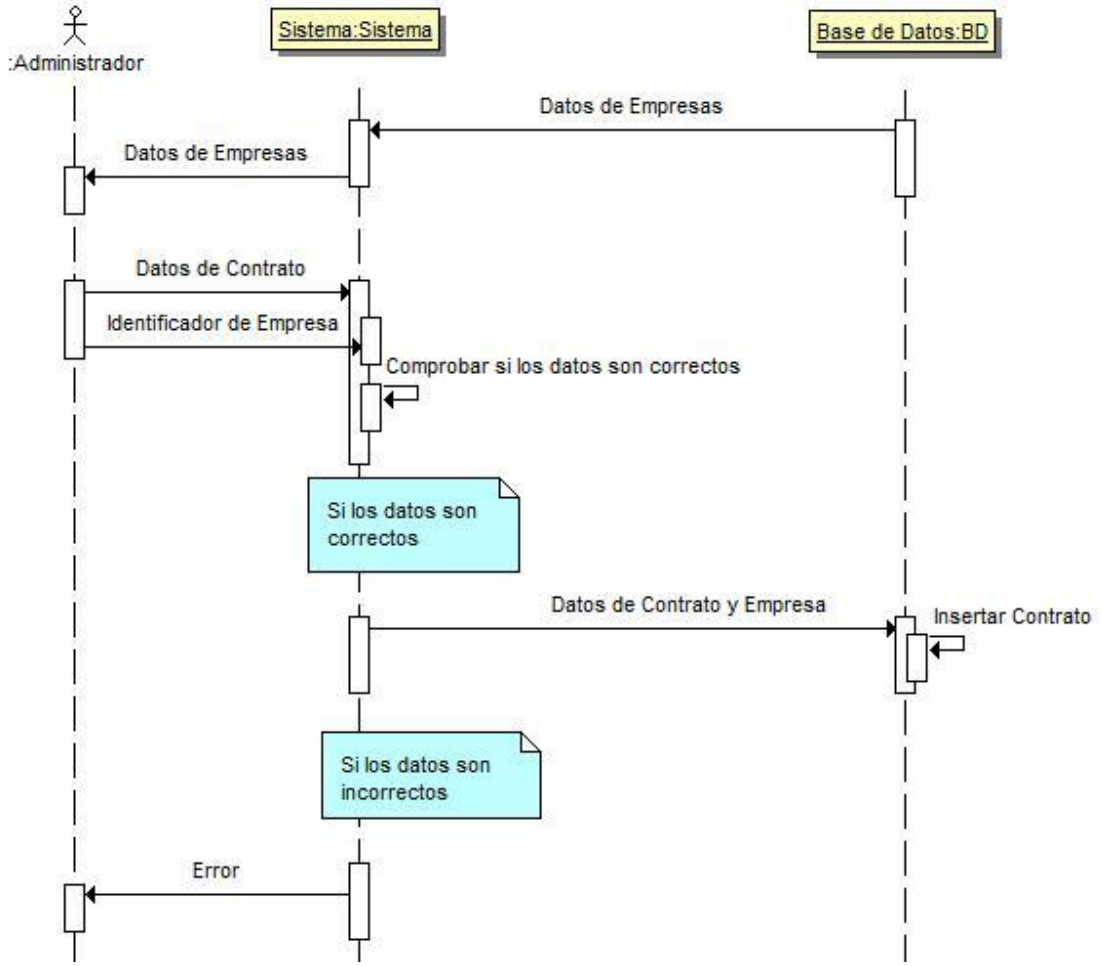


Figura 26: Diagrama de Secuencia, Insertar Contrato

CREAR COPIA BASE DATOS

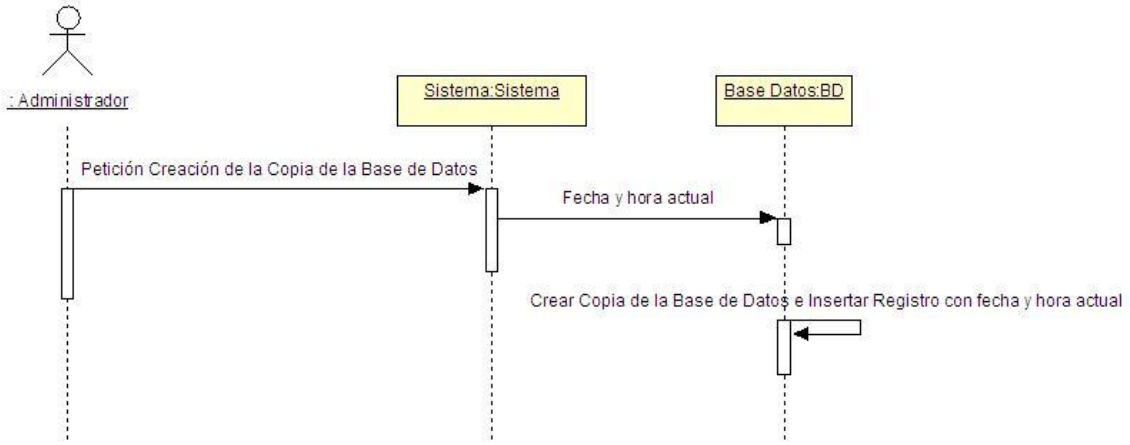


Figura 27: Diagrama de Secuencia, Crear Copia Base Datos

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INSERTAR FACTURA DESDE ARCHIVO XML

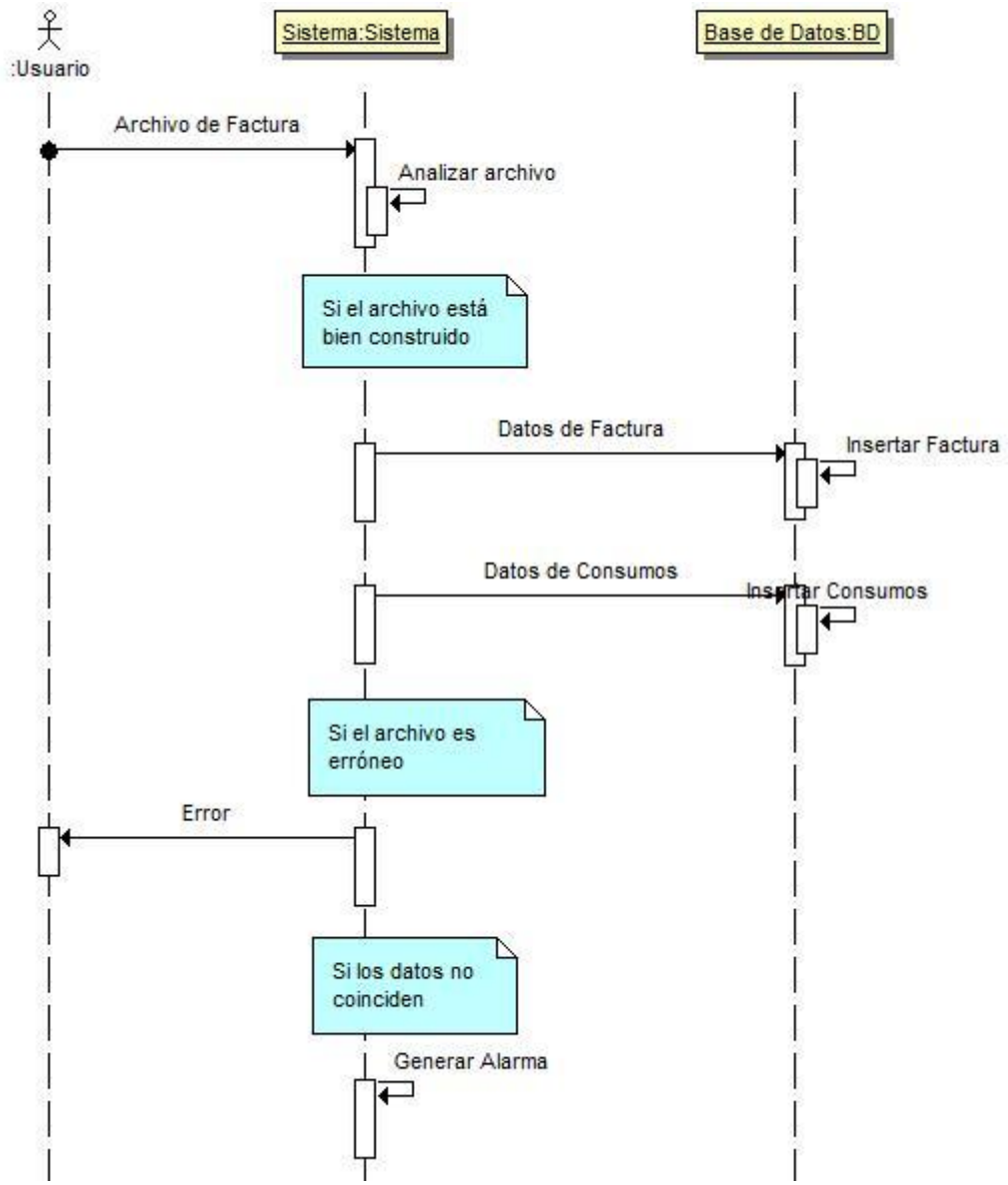


Figura 28: Diagrama de Secuencia, Insertar Factura desde Archivo XML

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INSERTAR FACTURAS DESDE ARCHIVO EXCEL

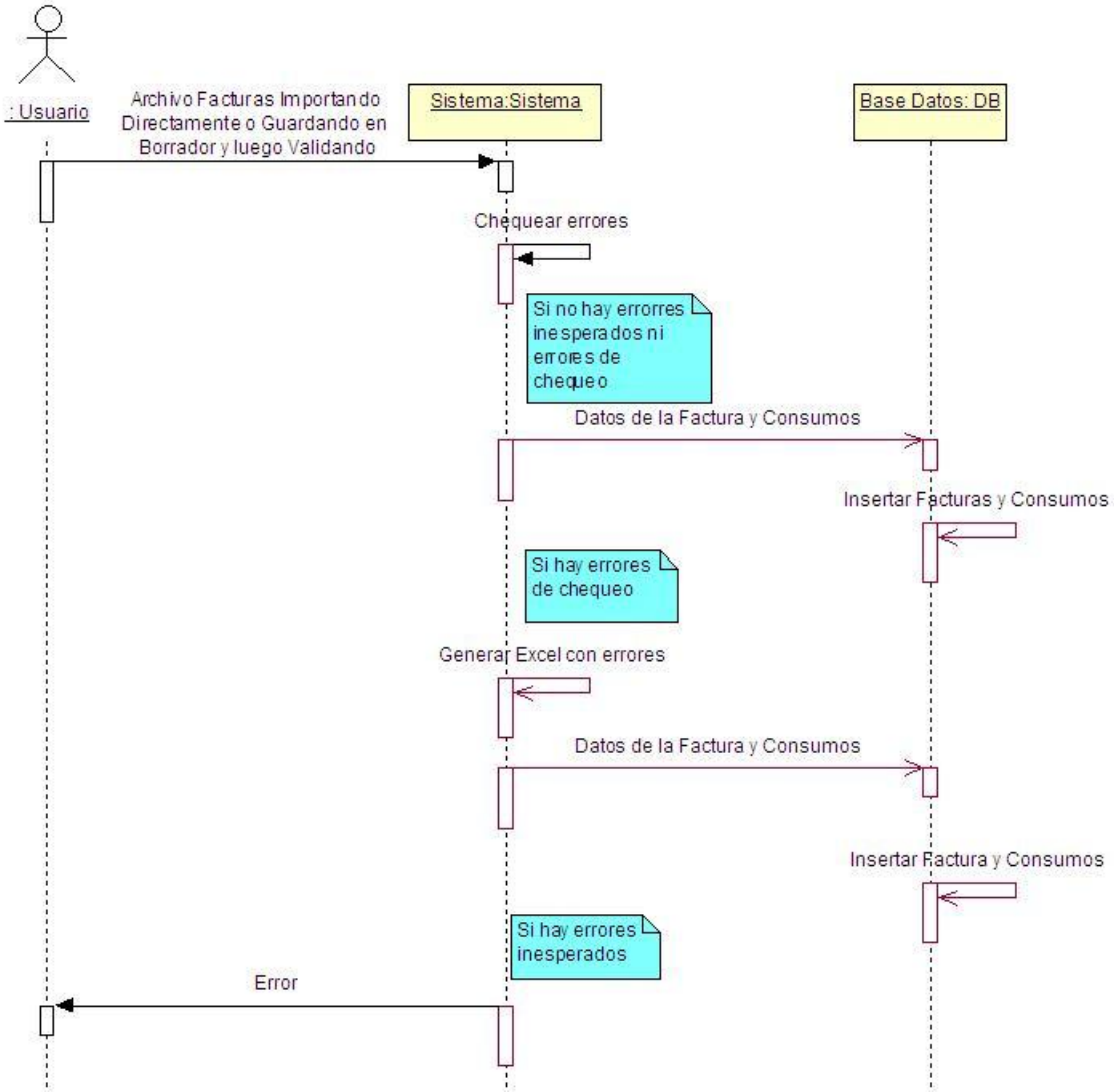


Figura 29: Diagrama de Secuencia, Insertar Facturas desde Archivo Excel

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INSERTAR FACTURA DESDE INTERFAZ

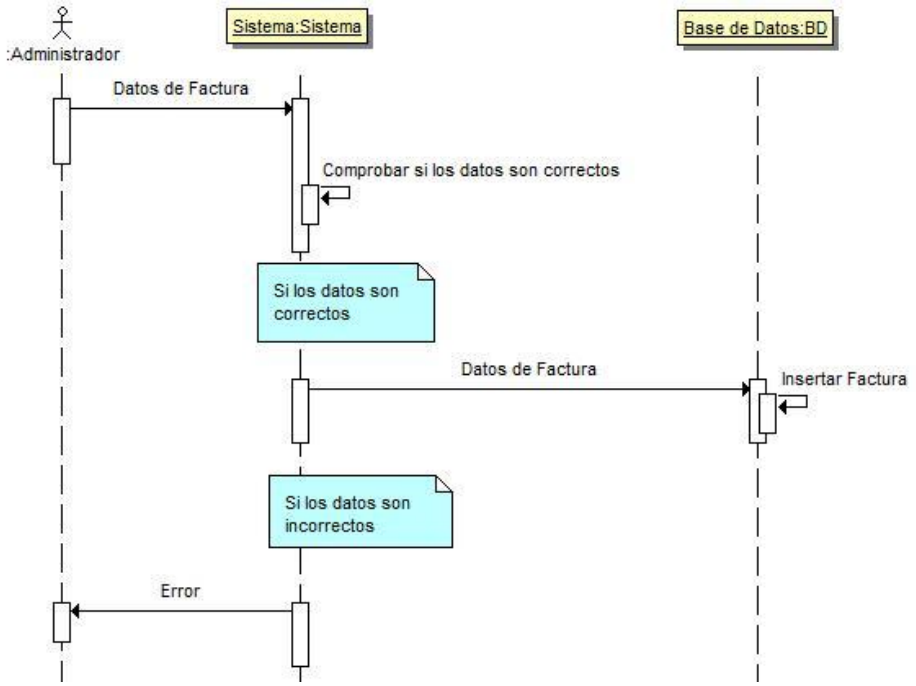


Figura 30: Diagrama de Secuencia, Insertar Factura desde Interfaz

MODIFICAR ORGANIZACIÓN

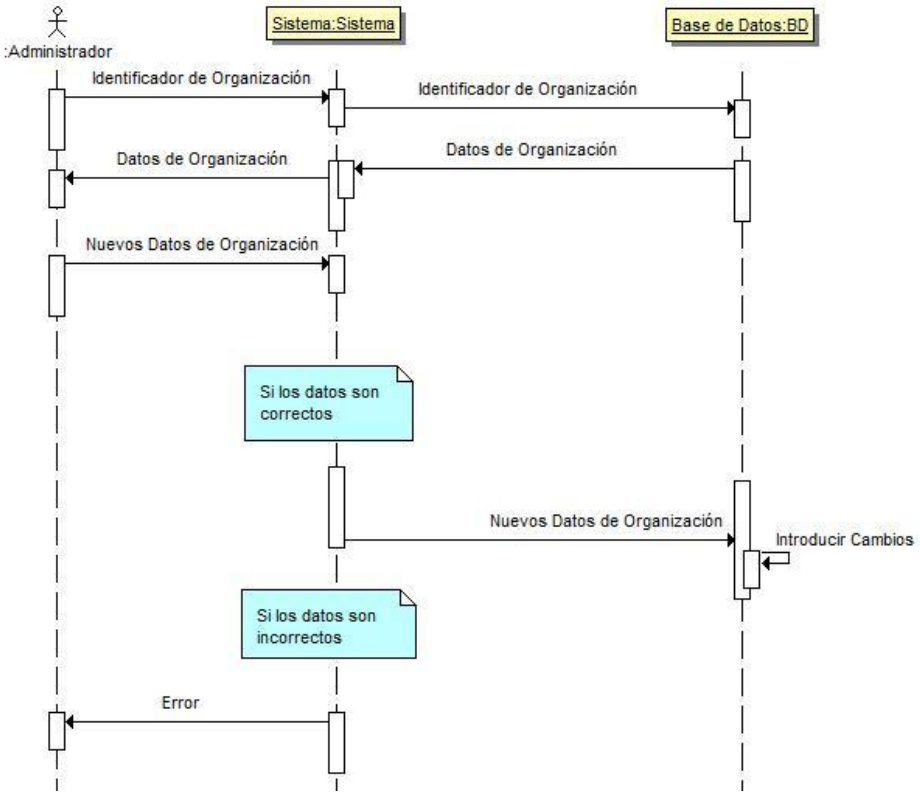


Figura 31: Diagrama de Secuencia, Modificar Organización

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MODIFICAR CUPS

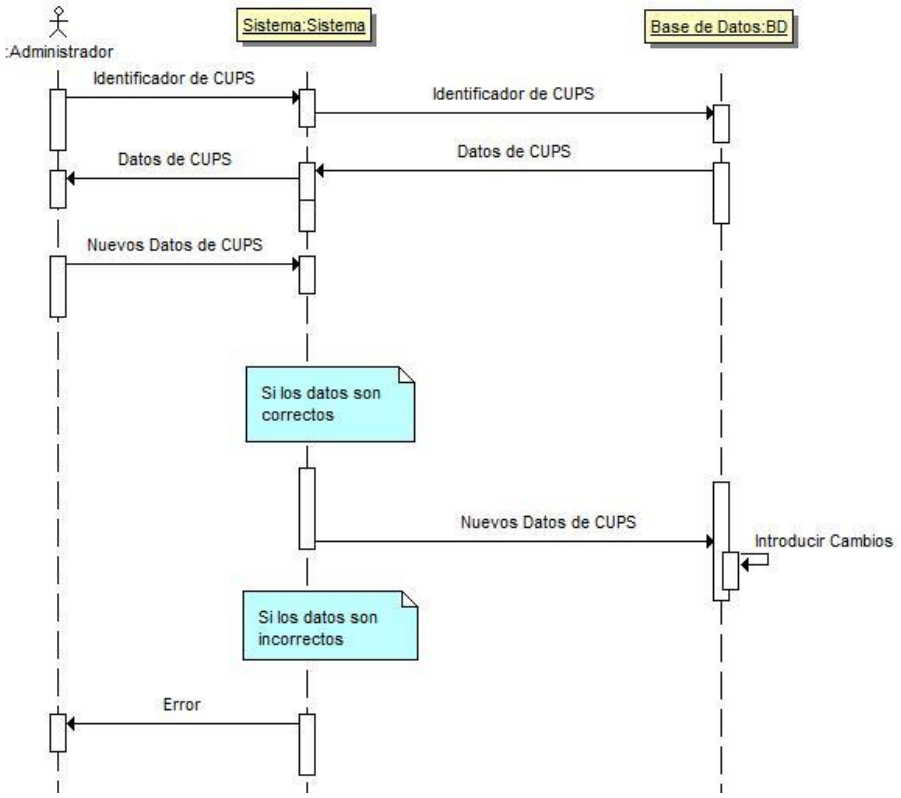


Figura 32: Diagrama de Secuencia, Modificar CUPS

MODIFICAR TARIFA

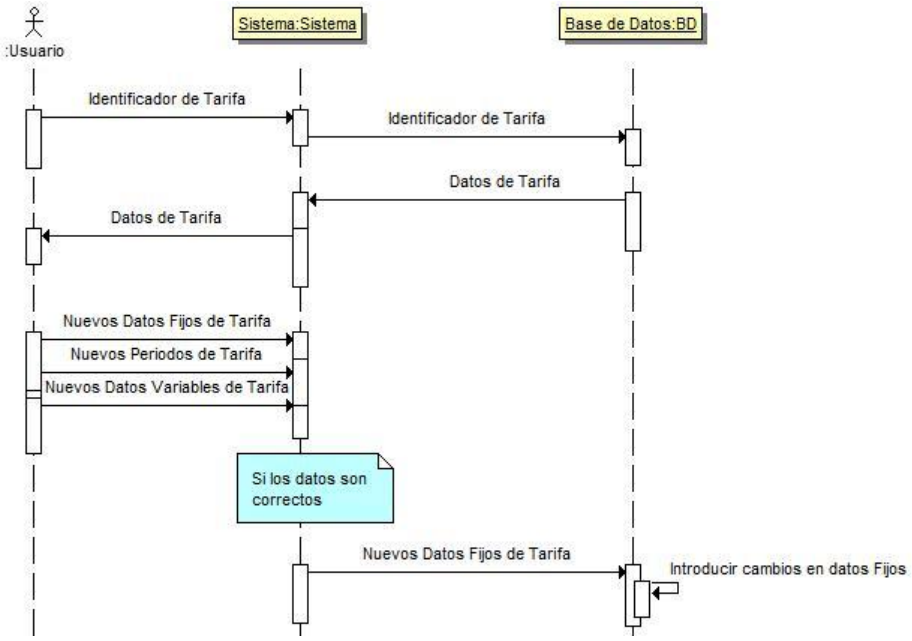


Figura 33: Diagrama de Secuencia, Modificar Tarifa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ACTUALIZAR PRECIOS TARIFA Y CONSULTA PRECIOS TARIFA

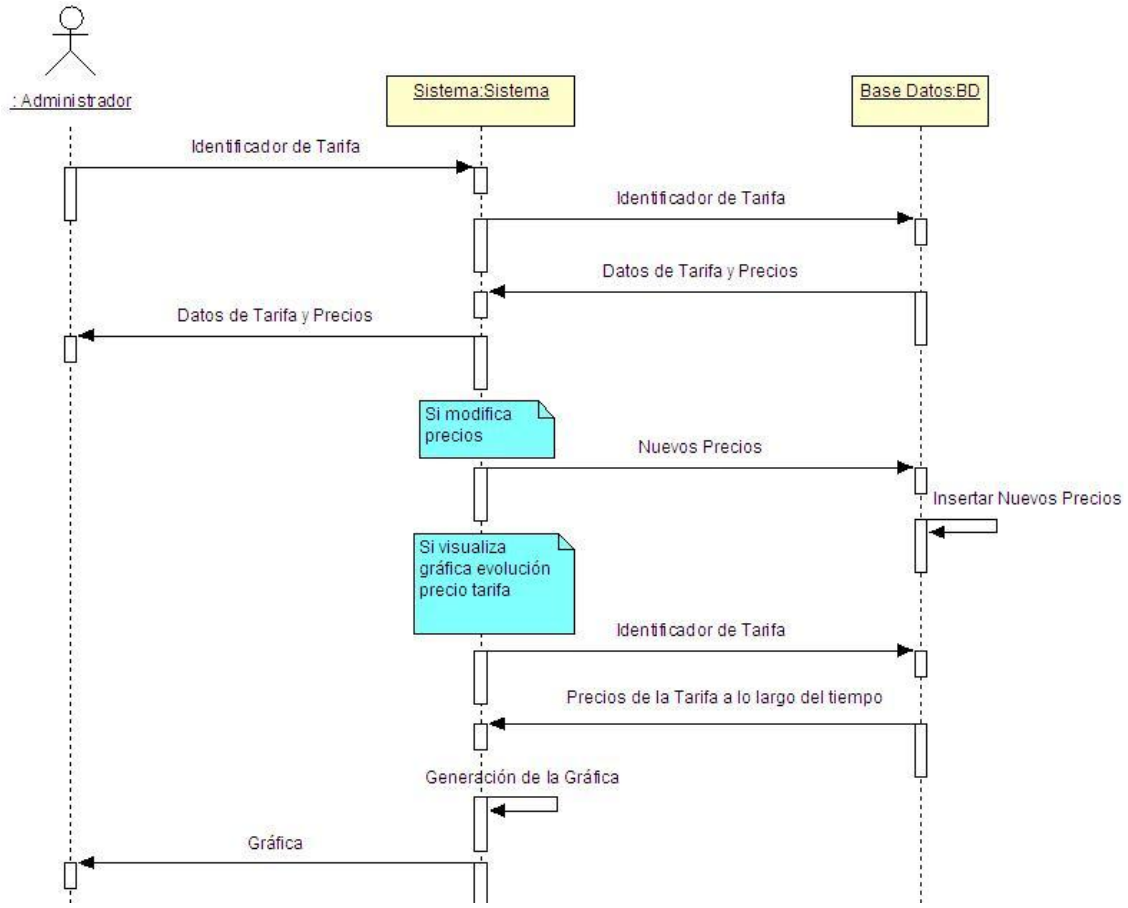


Figura 34: Diagrama de Secuencia, Actualizar Precios Tarifa y Consulta Precios Tarifa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MODIFICAR EMPRESA

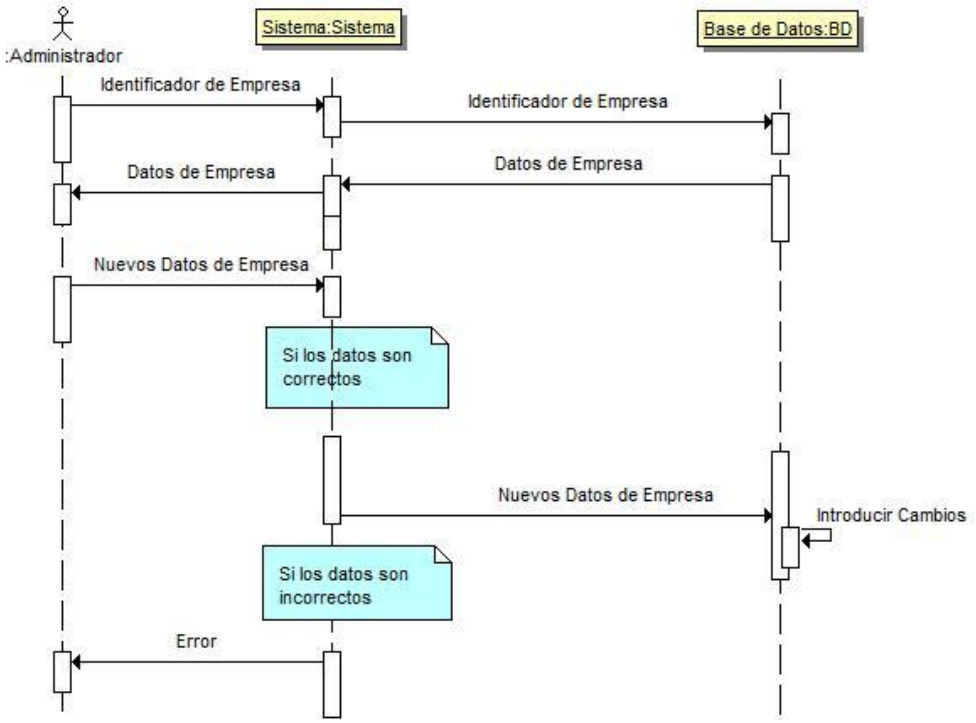


Figura 35: Diagrama de Secuencia, Modificar Empresa

MODIFICAR CONTRATO

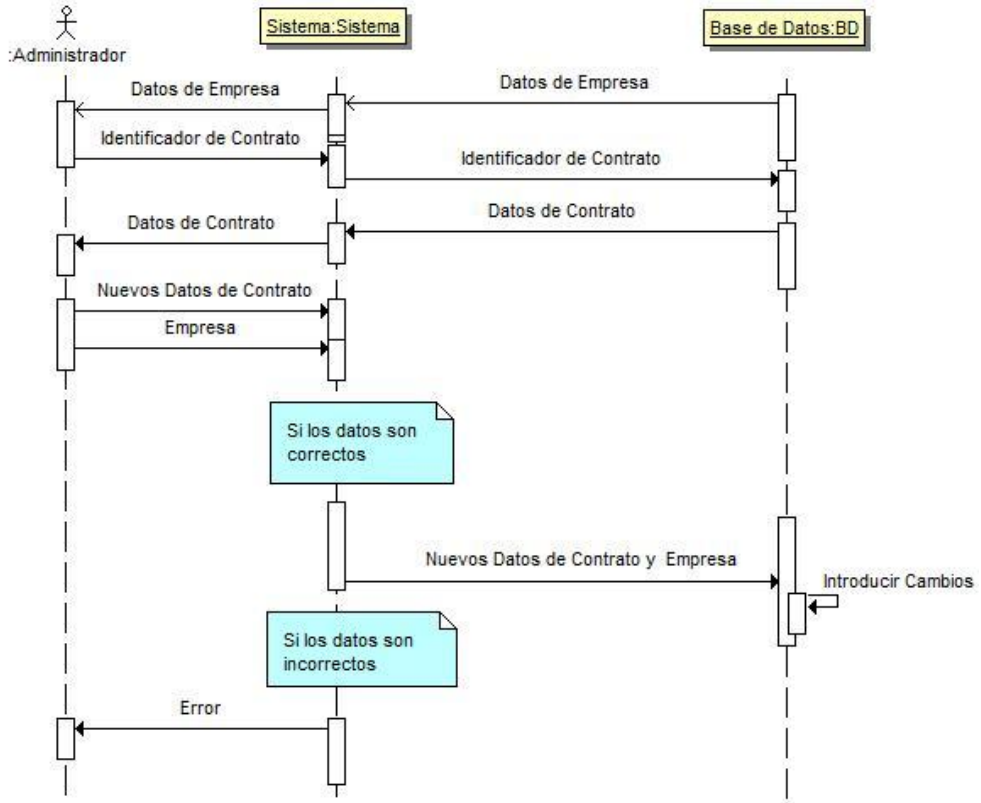


Figura 36: Diagrama de Secuencia, Modificar Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

MODIFICAR FACTURA

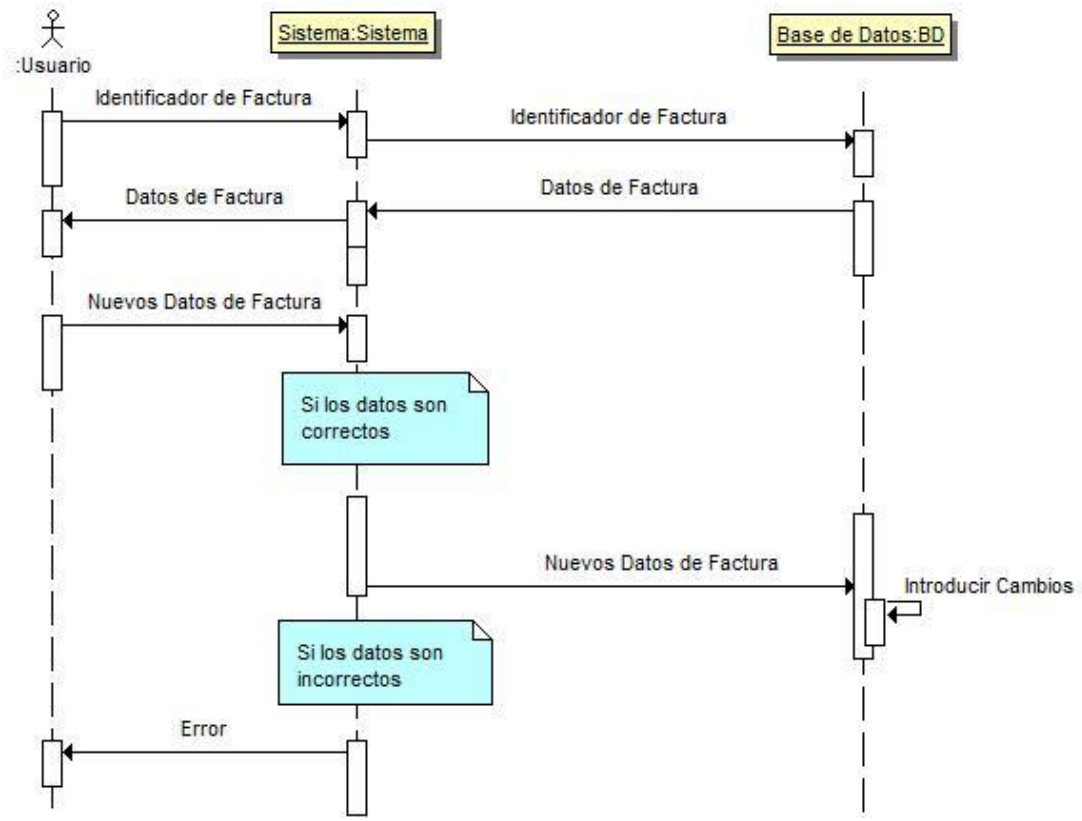


Figura 37: Diagrama de Secuencia, Modificar Factura

DAR DE BAJA CUPS

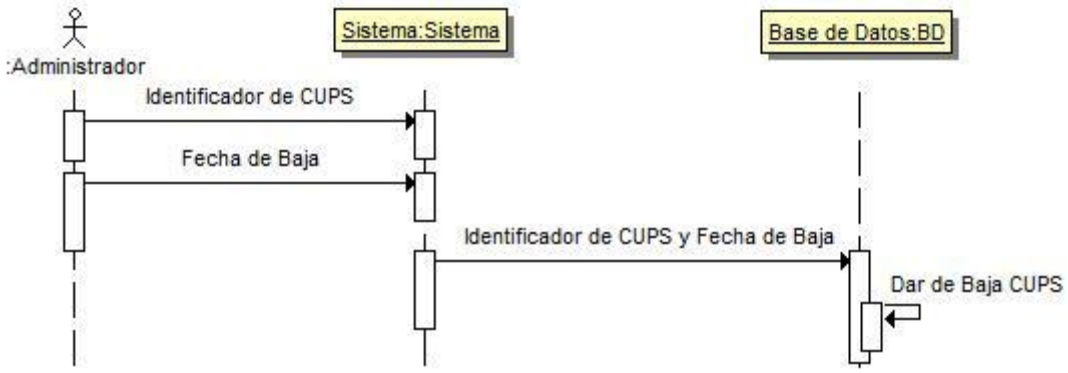


Figura 38: Diagrama de Secuencia, Dar de Baja CUPS

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DAR DE BAJA CONTRATO

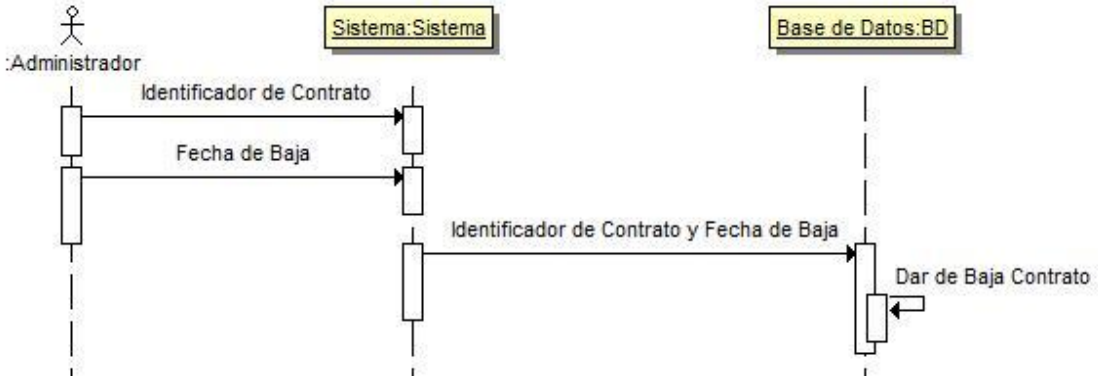


Figura 39: Diagrama de Secuencia, Dar de Baja Contrato

ASIGNAR CUPS A ORGANIZACIÓN

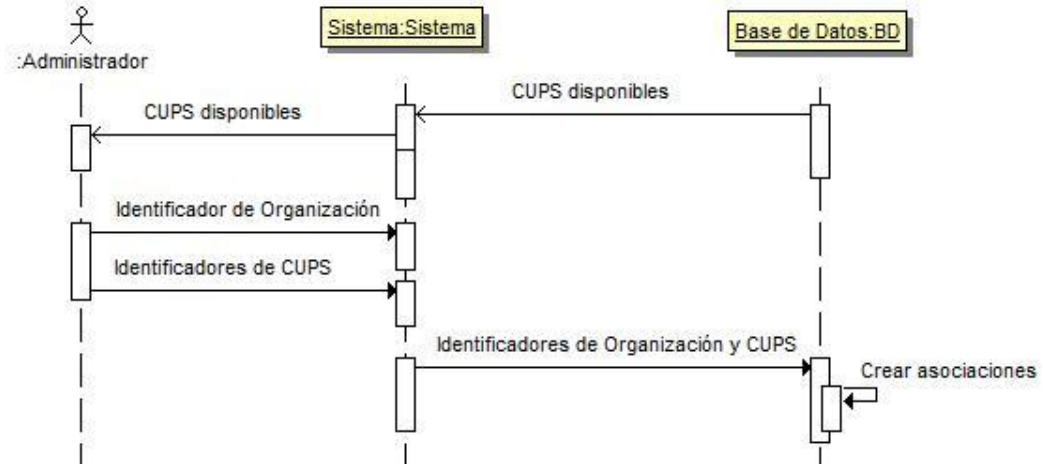


Figura 40: Diagrama de Secuencia, Asignar CUPS a Organización

REESTRUCTURAR ORGANIZACIÓN

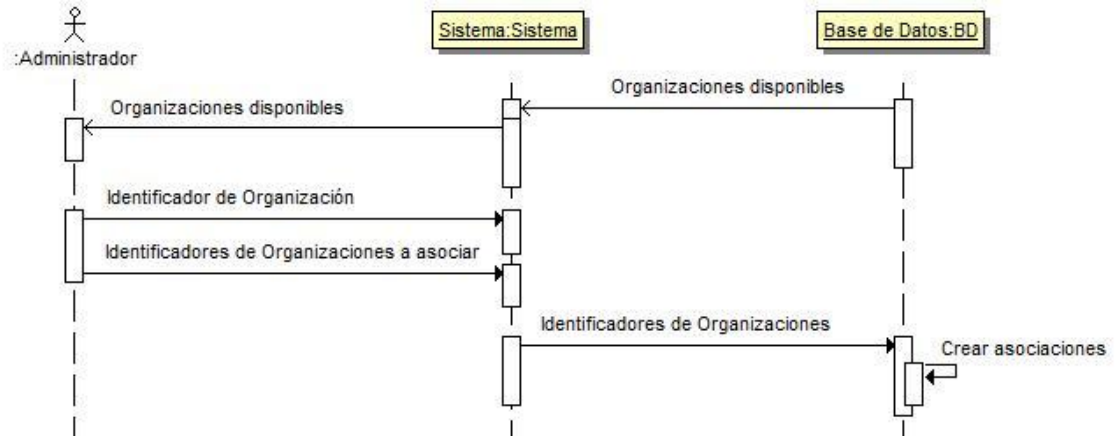


Figura 41 Reestructurar Organización

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ASIGNAR CUPS A CONTRATO

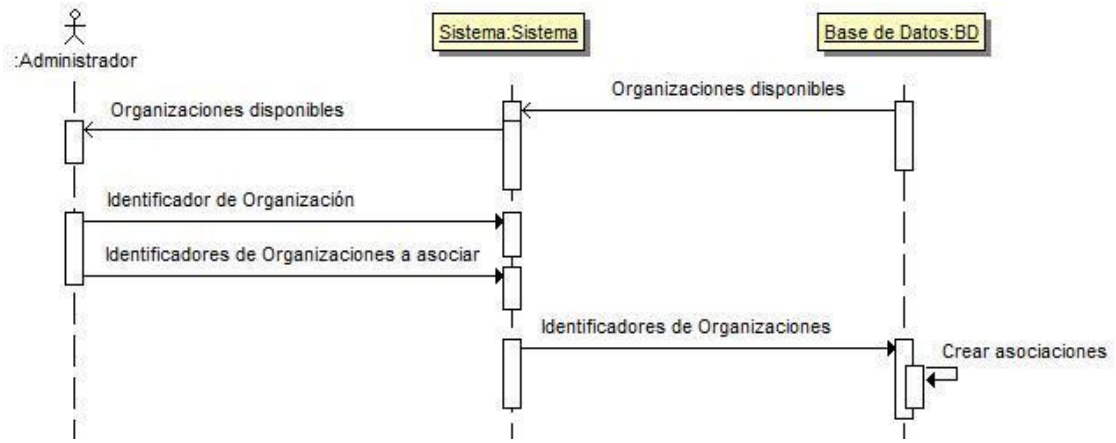


Figura 42: Diagrama de Secuencia, Asignar CUPS a Contrato

COMPROBAR FACTURA

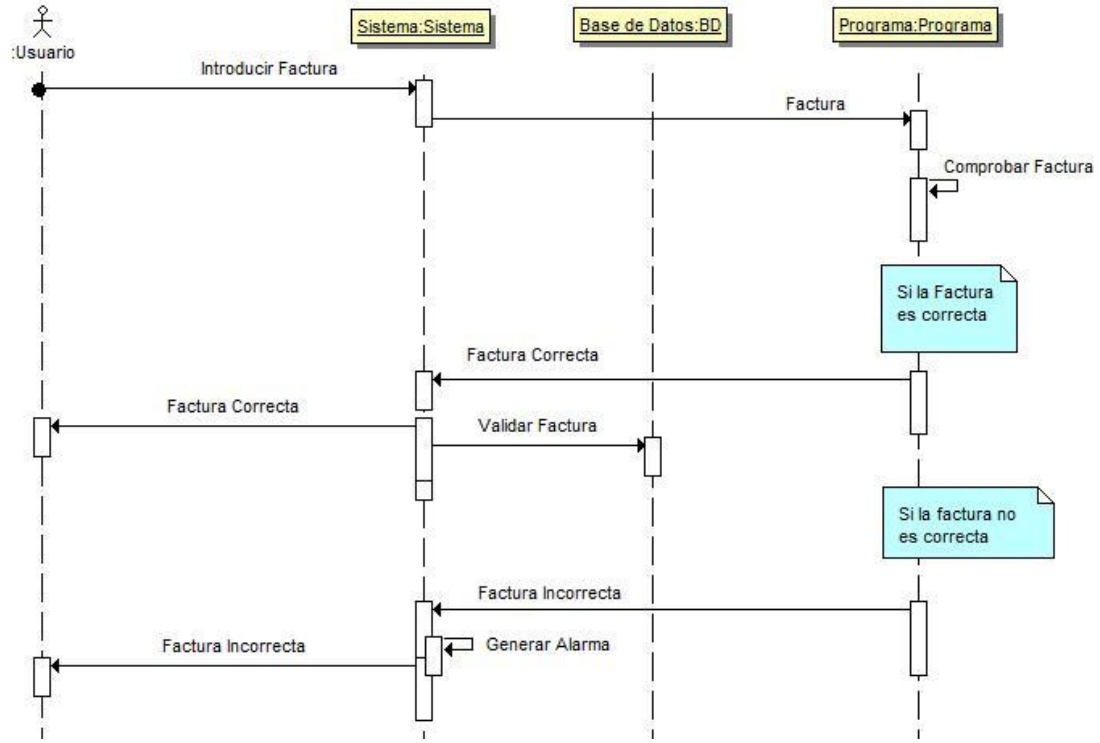


Figura 43: Diagrama de Secuencia, Comprobar Factura

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ACTIVAR ALARMA

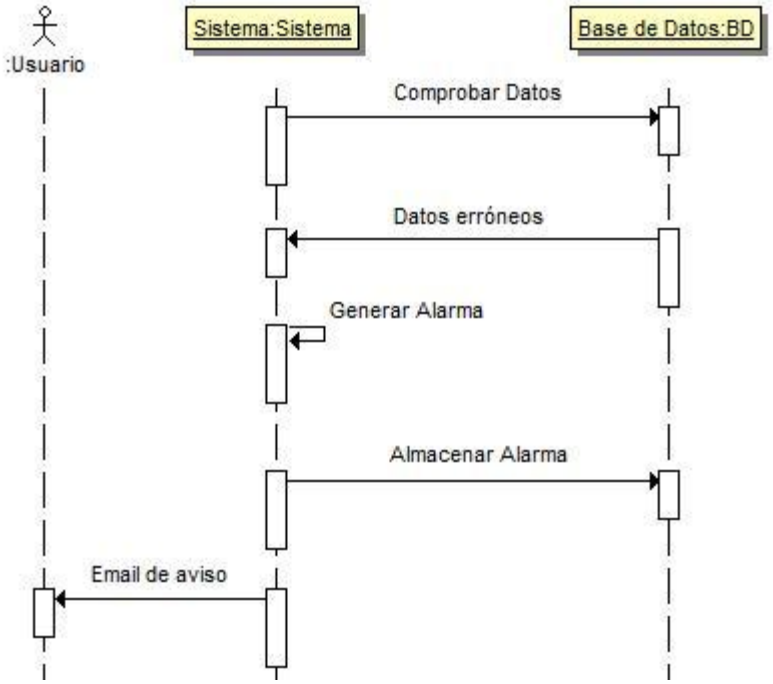


Figura 44: Diagrama de Secuencia, Activar Alarma

INFORME CONSUMO ACUMULADO

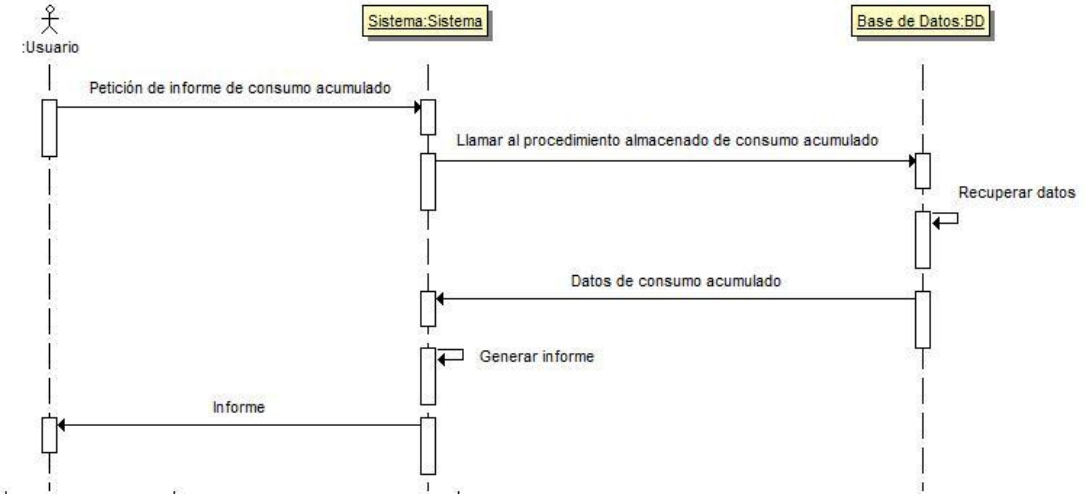


Figura 45: Diagrama de Secuencia, Informe Consumo Acumulado

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INFORME DE ENERGÍA ACTIVA

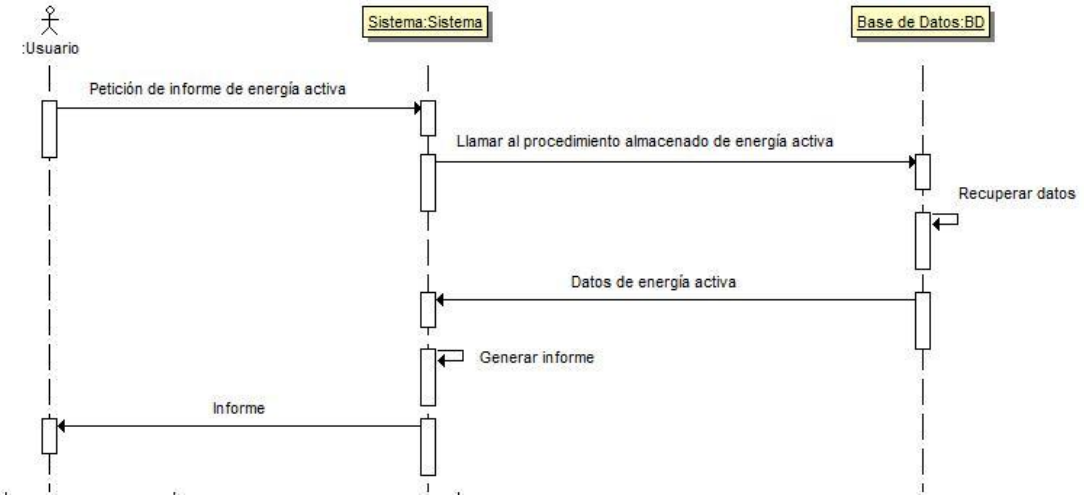


Figura 46: Diagrama de Secuencia, Informe de Energía Activa

INFORME DE GASTO TOTAL

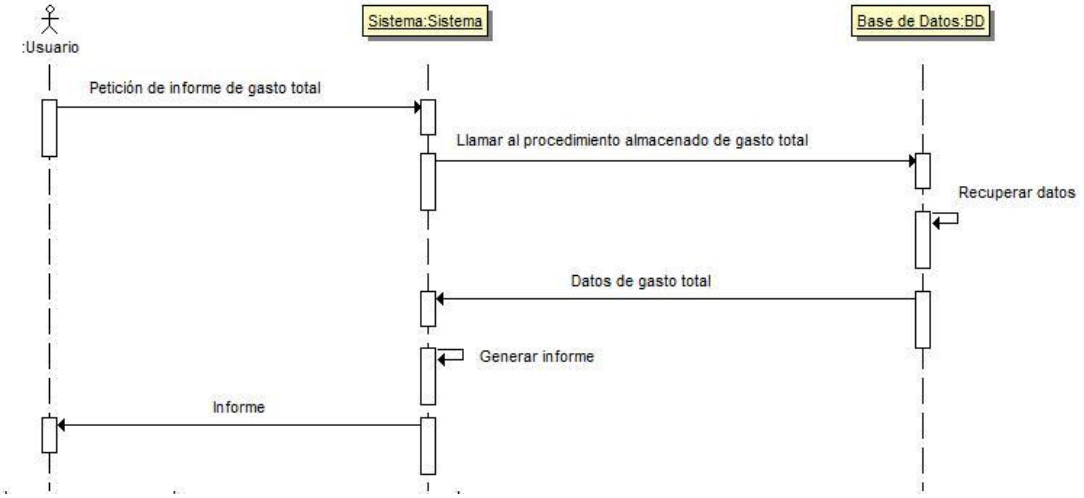


Figura 47: Diagrama de Secuencia, Informe de Gasto Total

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INFORME DE ENERGÍA REACTIVA

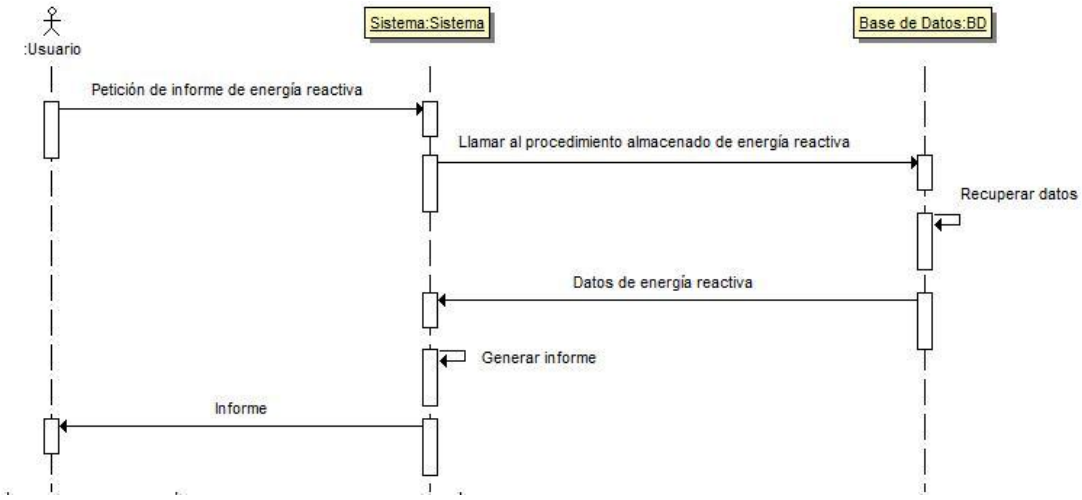


Figura 48: Diagrama de Secuencia, Informe de Energía Reactiva

INFORME DE MÁXIMO DE ENERGÍA ACTIVA

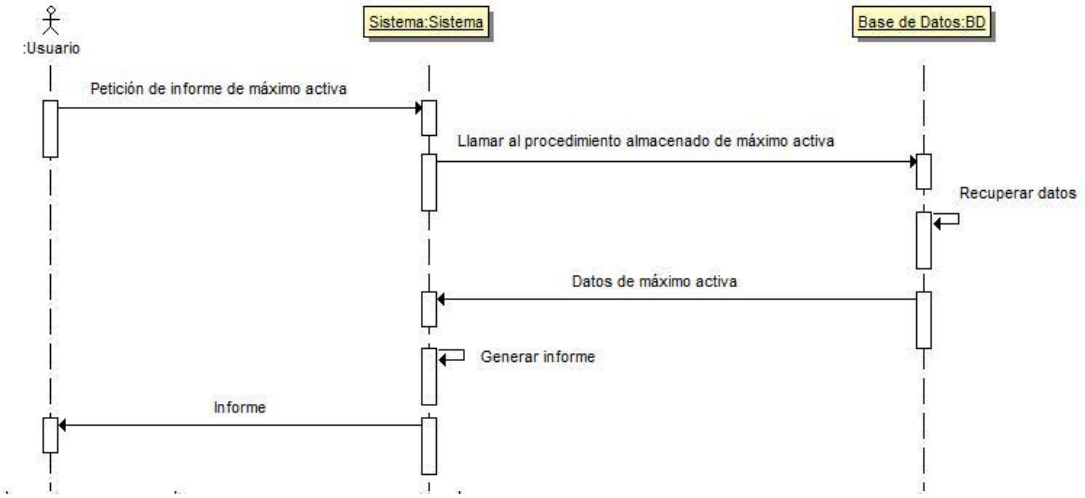


Figura 49: Diagrama de Secuencia, Informe de Máximo de Energía Activa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INFORME DE EXCESO DE POTENCIA

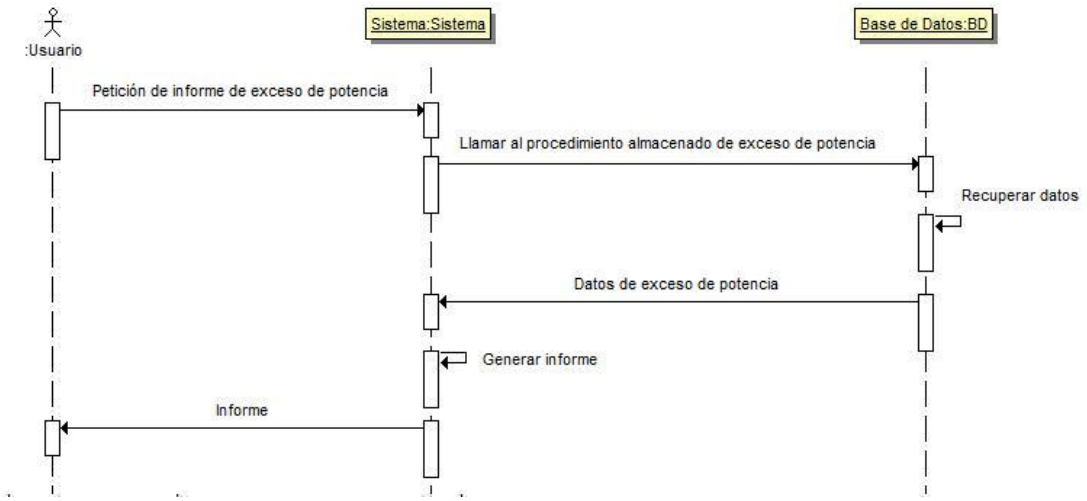


Figura 50: Diagrama de Secuencia, Informe de Exceso de Potencia

INFORME DE GASTO POR ALQUILER DE EQUIPOS

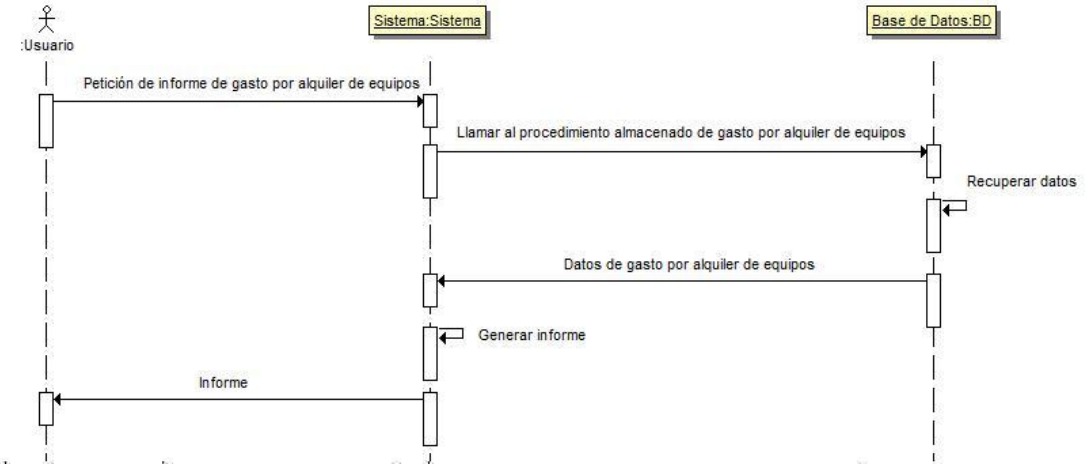


Figura 51: Diagrama de Secuencia, Informe de Gasto por Alquiler de Equipos

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INFORME DE PRODUCCIÓN DE CO_2

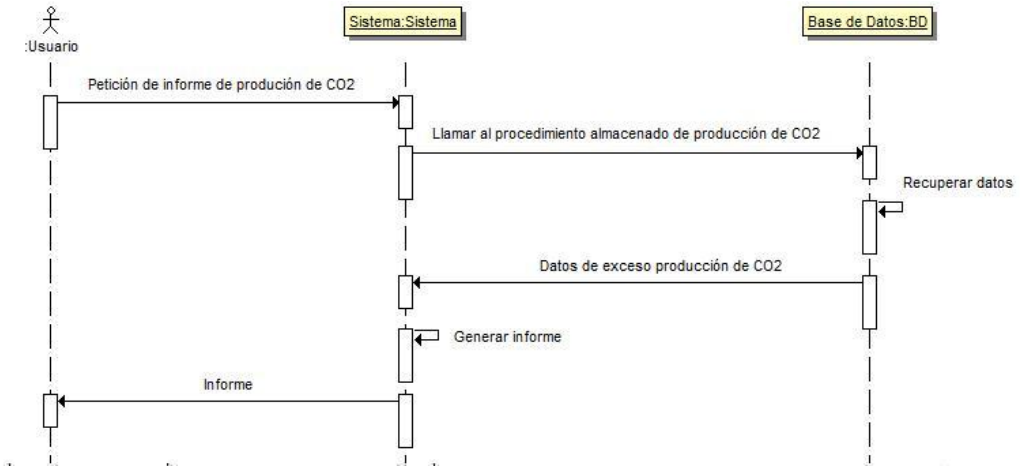


Figura 52: Diagrama de Secuencia, Informe de Producción de CO2

INFORME SUMINISTROS SOCORRO

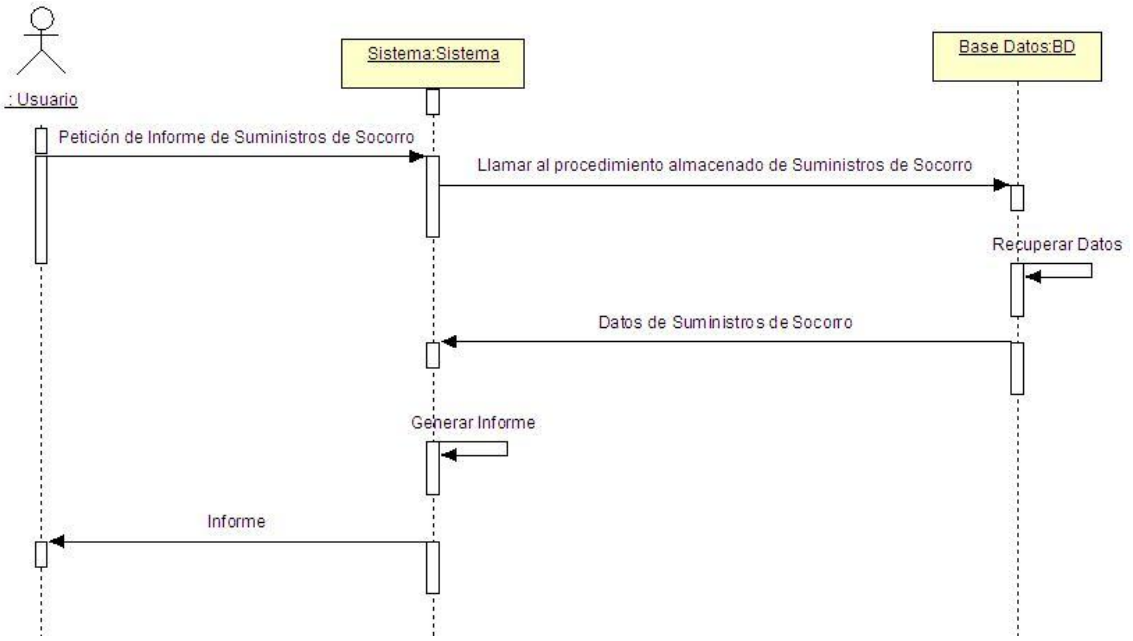


Figura 53: Diagrama de Secuencia, Informe Suministros Socorro

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INFORME CONTROL FACTURAS

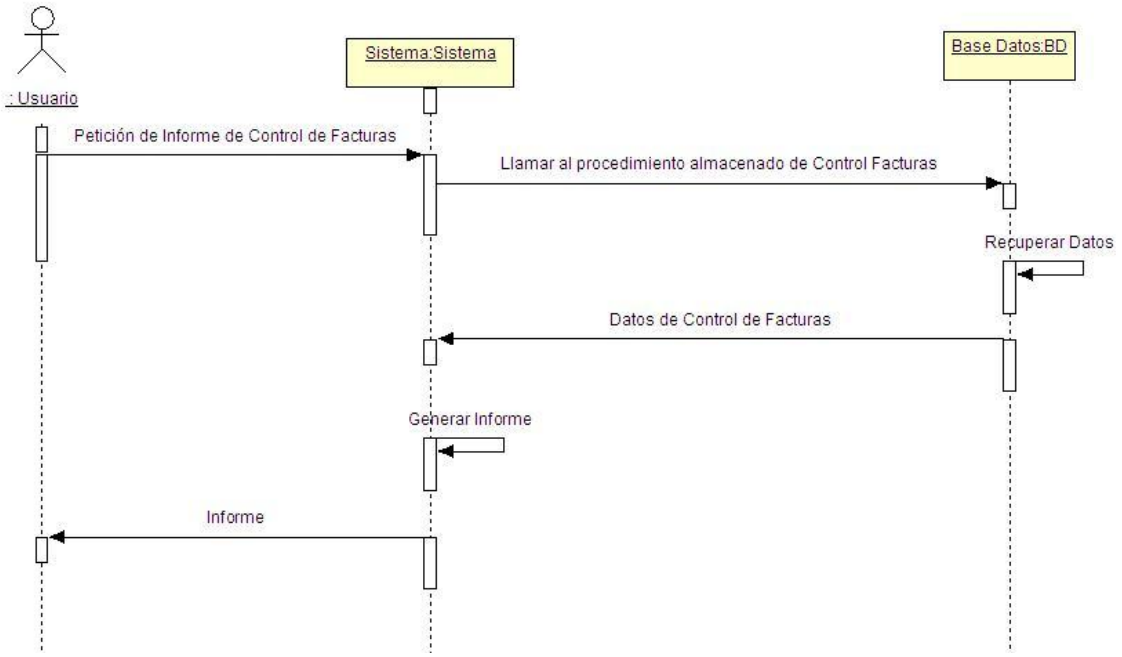


Figura 54: Diagrama de Secuencia, Informe Control Facturas

SIMULACIÓN DE CAMBIO DE POTENCIA

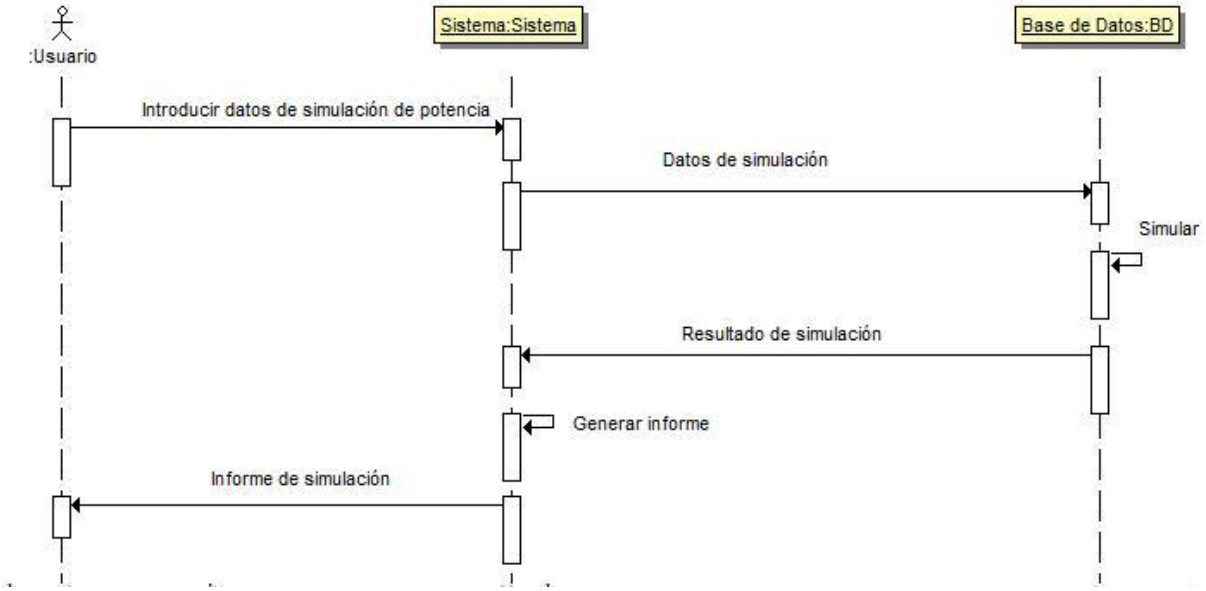


Figura 55: Diagrama de Secuencia, Simulación de Cambio de Potencia

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

SIMULACIÓN DE CAMBIO DE TARIFA

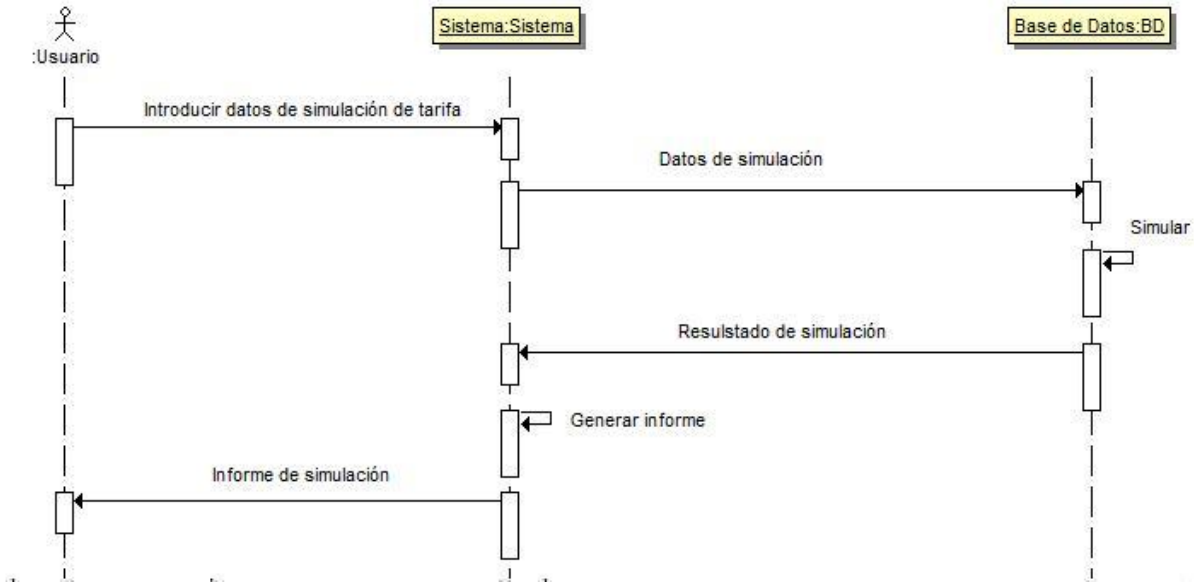


Figura 56: Diagrama de Secuencia, Simulación de Cambio de Tarifa

CREAR ALARMA

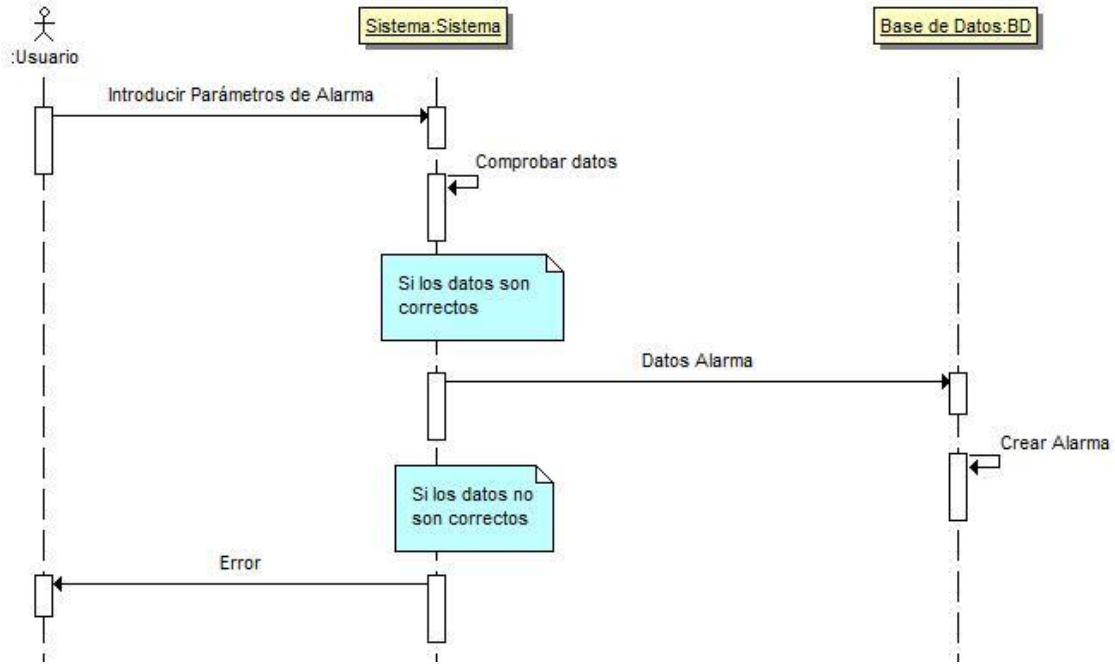


Figura 57: Diagrama de Secuencia, Crear Alarma

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DESHABILITAR ALARMA

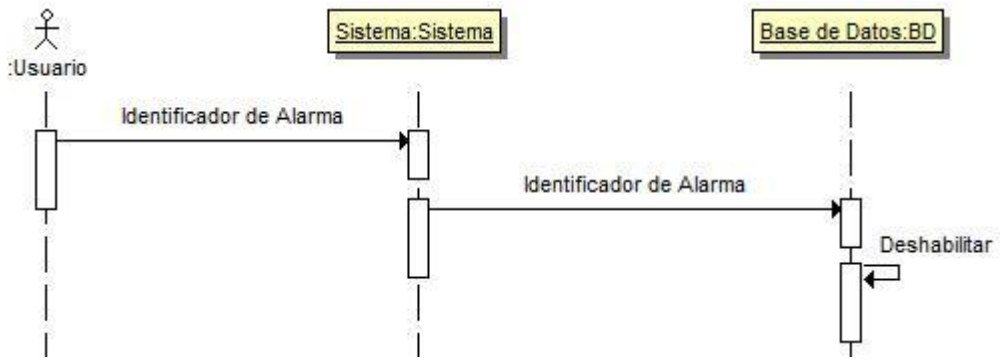


Figura 58: Diagrama de Secuencia, Deshabilitar Alarma

HABILITAR ALARMA

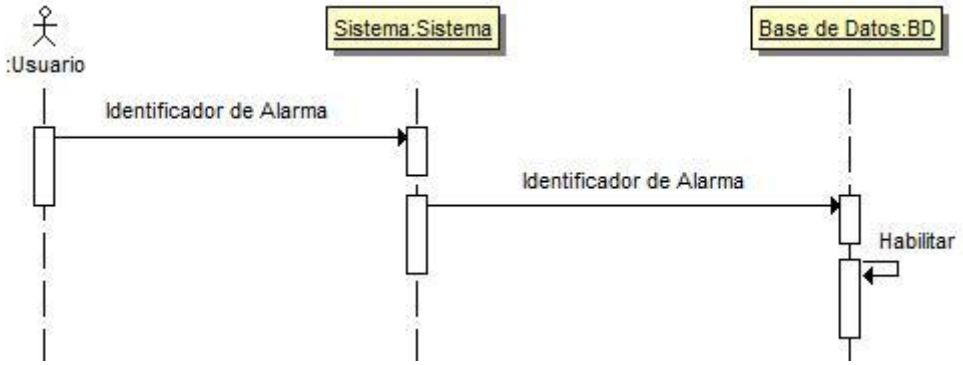


Figura 59: Diagrama de Secuencia, Habilitar Alarma

ELIMINAR ALARMA

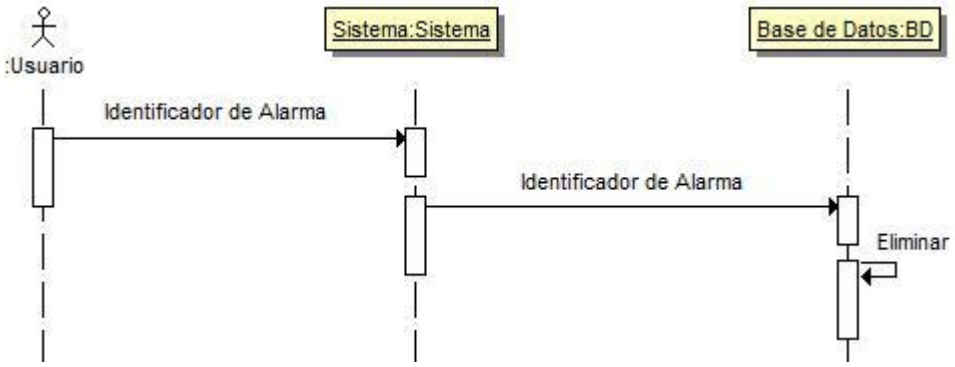


Figura 60: Diagrama de Secuencia, Eliminar Alarma

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONSULTAR ALARMAS

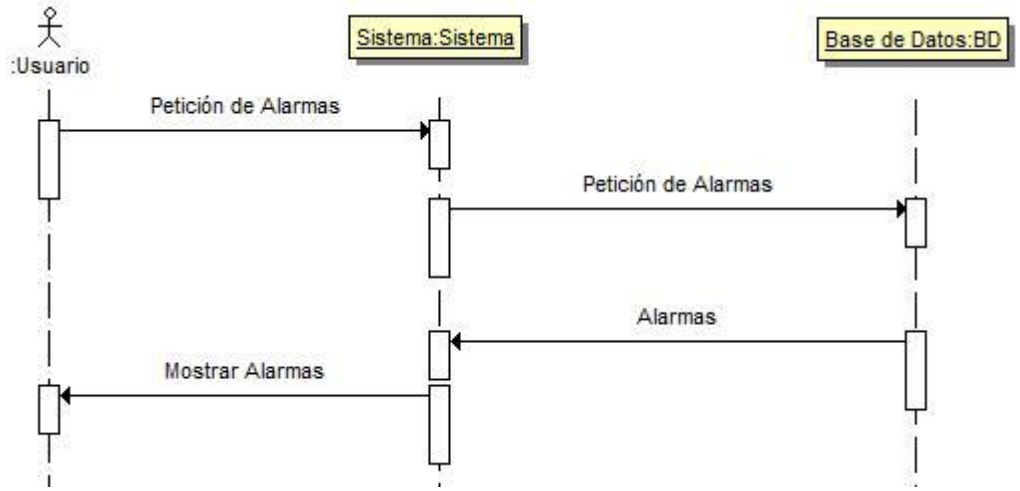


Figura 61: Diagrama de Secuencia, Consultar Alarmas

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

3.2 DIAGRAMA DE CLASES

Para ayudar a implementar los casos de uso y acorde con la Base de Datos, se obtiene el siguiente diagrama de clases del sistema. Cada clase está descrita con más detalle en una tabla tras el diagrama.

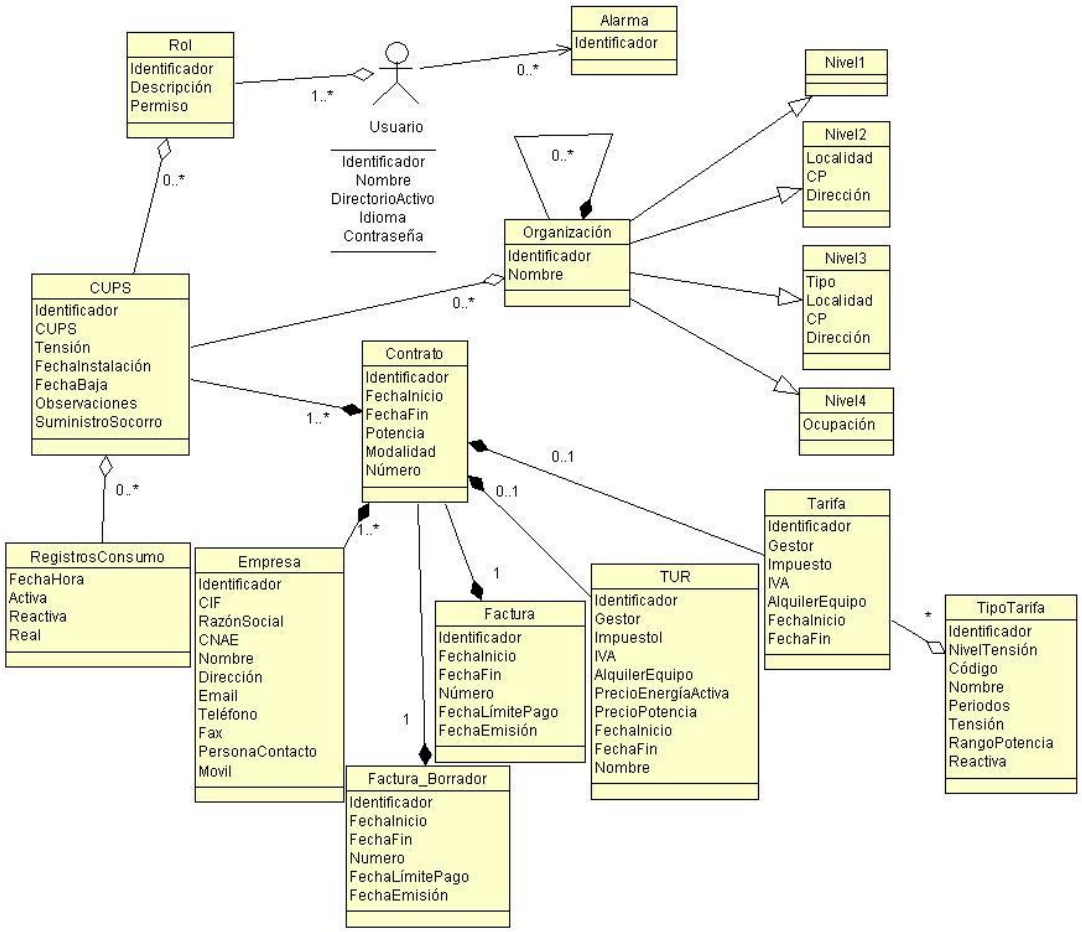


Figura 62: Diagrama de Clases

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Nombre: Usuario
Descripción: Usuario con acceso al sistema
Atributos: • Identificador, entero. Clave única. • Nombre, cadena. • DirectorioActivo, cadena. Para el acceso automático al SNS-O • Idioma, cadena. Idioma predeterminado en el que accederá al sistema. • Contraseña, cadena cifrada. Clave personal de acceso al sistema.
Relaciones: • Un usuario tiene uno o más roles para poder acceder al sistema. Relación de agregación. • Un usuario puede estar relacionado con una o más alarmas. Cuando una de estas alarmas sea activada, recibirá una notificación. Relación de asociación.

Tabla 49: Clase Usuario

Nombre: Alarma
Descripción: Alarma creada por el usuario
Atributos: • Identificador, entero. Clave única.
Relaciones:

Tabla 50: Clase Alarma

Nombre: Rol
Descripción: Rol que tiene el usuario
Atributos: • Identificador, entero. Clave única. • Descripción, cadena. • Permiso, cadena. El perfil con el que el usuario entra. Puede tener los valores “Administrador”, “Datos” o “Consulta”.
Relaciones: • Un rol puede o no tener CUPS a los que puede acceder y consultar. Relación de agregación.

Tabla 51: Clase Rol

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Nombre: CUPS
Descripción: Código Único de Punto de Suministro
Atributos: • Identificador, entero. Clave única. • CUPS, cadena. Código CUPS. • Tensión, numérico. • FechaInstalación, fecha. • FechaBaja, fecha. • Observaciones, cadena. Notas para ayudar al usuario. • Suministros socorro, booleano. Si el CUPS es un suministro de socorro o no.
Relaciones: • Un CUPS tendrá un gran número de RegistrosConsumo. Relación de agregación.

Tabla 52: Clase CUPS

Nombre: RegistrosConsumo
Descripción: Consumos generados por un CUPS
Atributos: • FechaHora, fecha-hora. Momento en el que se ha registrado el consumo. • Activa, numérico. Consumo de energía activa. • Reactiva, numérico. Consumo de energía reactiva. • Real, booleano. Si la lectura es o no una lectura real.
Relaciones:

Tabla 53: Clase RegistrosConsumo

Nombre: Organización
Descripción: Organización del Servicio Navarro de Salud
Atributos: • Identificador, entero. Clave única. • Nombre, cadena. Nombre de la organización.
Relaciones: • Una Organización estará compuesta por otras organizaciones. Relación de composición. • Una organización puede ser de distintos tipos: Nivel1, Nivel2, Nivel3 o Nivel4. Relación de generalización. • Una organización puede o no tener CUPS. Relación de agregación.

Tabla 54: Clase Organización

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Nombre: Nivel 1
Descripción: Nivel más alto de la jerarquía del SNS-O
Atributos:
Relaciones:

Tabla 55: Clase Nivel 1

Nombre: Nivel 2
Descripción: Segundo nivel en la jerarquía del SNS-O
Atributos: • Localidad, cadena. • CP, cadena. Código Postal. • Dirección, cadena.
Relaciones:

Tabla 56: Clase Nivel 2

Nombre: Nivel 3
Descripción: Tercer nivel en la jerarquía del SNS-O
Atributos: • Tipo, cadena. Tipo de centro que representa. • Localidad, cadena. • CP, cadena. Código Postal. • Dirección, cadena.
Relaciones:

Tabla 57: Clase Nivel 3

Nombre: Nivel 4
Descripción: Nivel más bajo en la jerarquía del SNS-O
Atributos: • Ocupación, numérico. Parte del nivel 3 que ocupa.
Relaciones:

Tabla 58: Clase Nivel 4

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Nombre: Contrato
Descripción: Contrato con una comercializadora energética
Atributos: • Identificador, entero. Clave única. • FechaInicio, fecha. Fecha de inicio del contrato. • FechaFin, fecha. Fecha de finalización del contrato. • Potencia, numérico. Potencia contratada. • Modalidad, cadena. Modalidad del contrato. • Número, cadena. Número de contrato.
Relaciones: • Un Contrato será para un CUPS y sólo para uno. Relación de agregación • Un contrato habrá sido contratado con una empresa y solo con una. Relación de composición. • Un contrato estará compuesto o no por una tarifa, pero en ausencia de ésta deberá estar compuesto por una Tarifa de Último Recurso. • Un contrato estará compuesto o no por una Tarifa de Último Recurso, pero en ausencia de ésta deberá estar compuesto por una tarifa.

Tabla 59: Clase Contrato

Nombre: Empresa
Descripción: Empresa comercializadora energética
Atributos: • Identificador, entero. Clave única. • CIF, cadena. Código de Identificación Fiscal. • RazónSocial, cadena. • DirecciónFiscal, cadena. • CNAE, cadena. Clasificación Nacional de Actividades Económicas. • Nombre, cadena. Nombre de la empresa. • Dirección, cadena. • Email, cadena. • Teléfono, cadena. • Fax, cadena. • PersonaContacto, cadena. Persona con la que se mantienen las comunicaciones. • Movil, numérico.
Relaciones:

Tabla 60: Clase Empresa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Nombre: Factura
Descripción: Factura eléctrica recibida de una compañía
Atributos: • Identificador, entero. Clave única. • FechaInicio, fecha. Fecha de inicio del periodo facturado. • FechaFin, fecha. Fecha de fin del periodo facturado. • Número, cadena. Número de factura. • FechaLímiteDePago, fecha. Último día en el que se podrá abonar la factura. • FechaEmisión, fecha. Día en el que la factura fue emitida.
Relaciones: • Una factura pertenece a un contrato y sólo a uno, aunque para cada contrato habrá muchas facturas.Relación de composición.

Tabla 61: Clase Factura

Nombre: Factura_Borrador
Descripción: Factura eléctrica recibida de una compañía, almacenada en borrador
Atributos: • Identificador, entero. Clave única. • FechaInicio, fecha. Fecha de inicio del periodo facturado. • FechaFin, fecha. Fecha de fin del periodo facturado. • Número, cadena. Número de factura. • FechaLímiteDePago, fecha. Último día en el que se podrá abonar la factura. • FechaEmisión, fecha. Día en el que la factura fue emitida.
Relaciones: • Una factura pertenece a un contrato y sólo a uno, aunque para cada contrato habrá muchas facturas.Relación de composición.

Tabla 62: Clase Factura_Borrador

Nombre: Tarifa
Descripción: Tarifa Energética
Atributos: • Identificador, entero. Clave única. • Gestor, cadena. Persona encargada de gestionar la tarifa. • Impuesto, numérico. Impuesto eléctrico. • IVA, numérico. Impuesto sobre el Valor Añadido. • AlquilerEquipo, numérico. Coste del alquiler de los equipos eléctricos. • FechaInicio, fecha. Día de inicio de la validez de la tarifa. • FechaFin, fecha. Día de finalización de la validez de la tarifa.
Relaciones:

Tabla 63: Clase Tarifa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Nombre: TipoTarifa
Descripción: Datos Fijos de una Tarifa
Atributos: • Identificador, entero. Clave única. • NivelTensión, numérico. Nivel de tensión al que se aplica la tarifa. • Código, cadena. • Nombre, cadena. • Periodos, entero. Número de periodos de la tarifa. • Tensión, numérico. • RangoPotencia, cadena. Rango donde se encuentra la potencia posible a contratar. • Reactiva, booleano. Si es para energía reactiva o no.
Relaciones: • Un tipo de tarifa no podrá variar, pero sí las tarifas. Por tanto un tipo de tarifa podrá tener varias tarifas.Relación de agregación.

Tabla 64: Clase TipoTarifa

Nombre: TUR
Descripción: Datos Fijos de una Tarifa
Atributos: • Identificador, entero. Clave única. • Gestor, cadena. Persona encargada de gestionar la tarifa. • Impuesto, numérico. Impuesto eléctrico. • IVA, numérico. Impuesto sobre el Valor Añadido. • AlquilerEquipo, numérico. Coste del alquiler de los equipos eléctricos. • PrecioPotencia, numérico. Precio de la potencia. • PrecioEnergíaActiva, numérico. Precio de la energía. • FechaInicio, fecha. Día de inicio de la validez de la tarifa. • FechaFin, fecha. Día de finalización de la validez de la tarifa. Nombre, cadena.
Relaciones:

Tabla 65: Clase TUR

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

3.3 DIAGRAMA DE LA BASE DE DATOS

Este es el diagrama relacional de la base de datos que finalmente ha resultado desde que se creó [1] con las posteriores modificaciones y cambios necesarios:

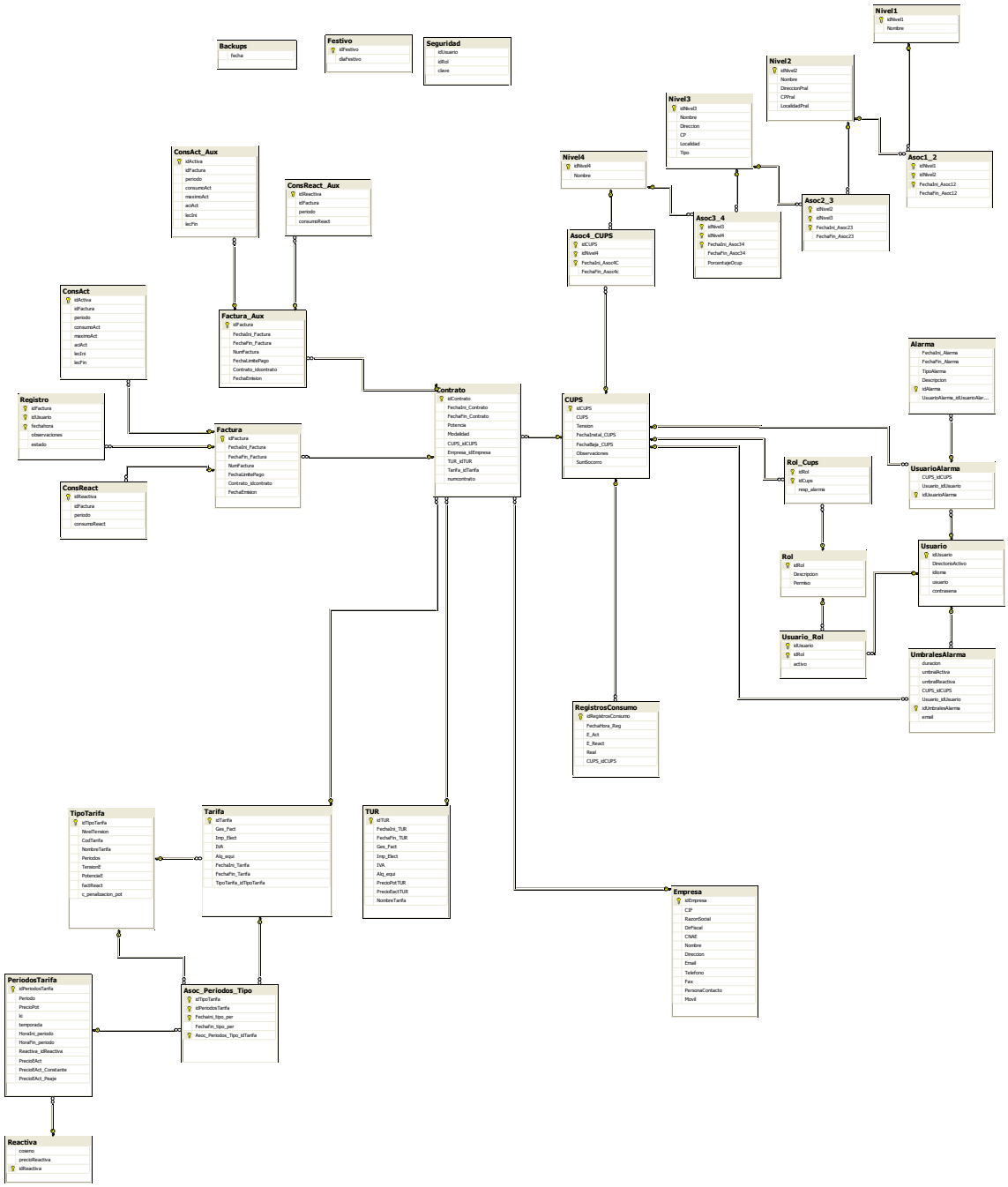


Figura 63: Diagrama de la Base de Datos

En un anexo final se mostrará este mismo diagrama más amplio para poder visualizarlo mejor y con tipos de datos.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

3.4 DISEÑO DE LA ARQUITECTURA DEL SISTEMA

Por los motivos comentados anteriormente en la sección de objetivos y análisis de requisitos, se ha optado por una aplicación web, optimizando así la posibilidad de acceso desde cualquier punto.

La web será una página en la que estará insertada la aplicación de Microsoft Silverlight encargada de la gestión del sistema. El sistema está formado por una arquitectura cliente-servidor tal y como se ve en la siguiente figura:

Figura 64: Arquitectura cliente - servidor

- **Usuario:** Las personas que utilizarán el sistema.
- **Silverlight:** Interfaz web desarrollada en Microsoft Silverlight.
- **Servidor web:** Servidor en el que estará alojada la web y la aplicación Silverlight, así como los servicios que sean necesarios.
- **Lógica de negocio:** Parte de la aplicación que se encargará de las tareas de gestión.
- **Datos eléctricos:** Datos sobre facturas, empresas, tarifas y consumos eléctricos, almacenados en una Base de Datos Microsoft SQLServer 2008.
- **Datos de gestión:** Datos sobre organizaciones y usuarios, almacenados en una Base de Datos Microsoft SQLServer 2008.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Además, la aplicación Silverlight tiene su propia arquitectura en capas, implementada así por sus desarrolladores con intención de aumentar la seguridad, otro motivo por el que es muy conveniente su uso en este proyecto. Otro de los motivos, obviando el ya mencionado de la compatibilidad con las plataformas ya existentes, es que es una plataforma totalmente gratuita.

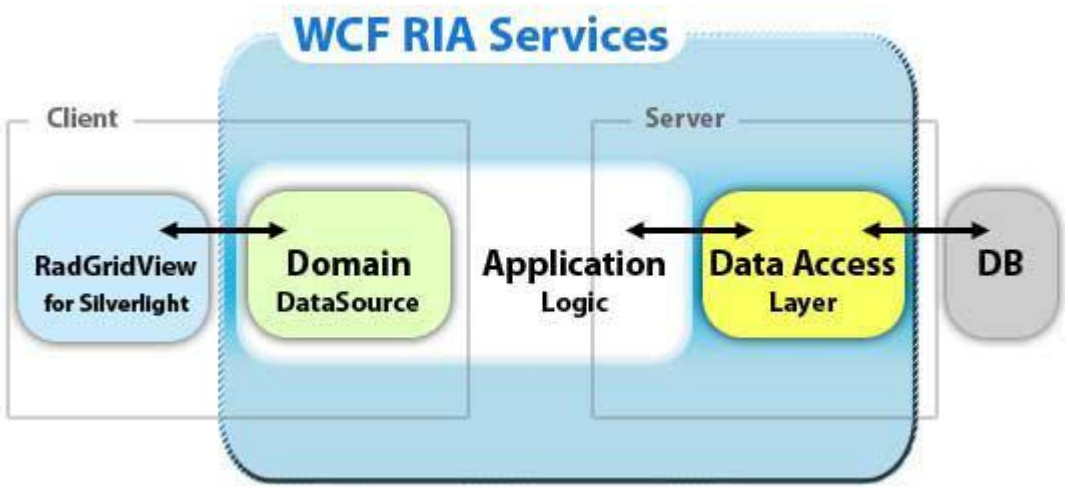


Figura 65: Arquitectura de Aplicaciones Silverlitght

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

IMPLEMENTACIÓN

Capítulo sobre la implementación del sistema, con detalles sobre los problemas encontrados y el modo de resolverlos

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4 IMPLEMENTACIÓN

4.1 PLATAFORMA DE DESARROLLO

En cualquier aplicación que sea de un tamaño grande, se necesita disponer de un IDE (Integrated Development Environment) que nos simplifique y sea más fácil su programación. Gracias al estar desarrollando la aplicación en una plataforma de Microsoft (Microsoft Silverlight), el programa que nos permite tanto poder realizar un buen diseño de las páginas (el interfaz) como todo el código que va por detrás es el Microsoft Visual Studio.

Se ha decidido usar concretamente el Microsoft Visual Studio 2010 ya que es el mismo que se usó para comenzar con el proyecto y es suficiente para continuar con el desarrollo de la aplicación. Dicho programa, permite compilar el código y generar los ejecutables (en nuestro caso complementos de la página web) de forma fácil.

Además permite conectarse con el servidor Internet Information Server de Microsoft, el cuál es el servidor que proporcionará finalmente las páginas web y accederá a los Servicios Web desarrollados en el sistema (necesarios para el acceso a la Base de Datos).

También el Visual Studio dispone de un Servidor web propio que se pone en marcha cada vez que se ejecute la aplicación en modo de pruebas, sin ser necesario de publicar la aplicación cuando se están realizando las pruebas.

Otra herramienta interesante es Microsoft Expression Blend 4, el cual permite diseñar las páginas web de un modo más visual, generando automáticamente el código .xaml necesario en respuesta a las acciones gráficas del programador.

Al igual que permite crear animaciones con estados de inicio y fin, siendo el programa el creador del código necesario para el movimiento de esas animaciones.

Instalados los IDE necesarios para la programación, compilación, ejecución y publicación de nuestras aplicaciones, aún no se está preparado para la programación en Microsoft Silverlight.

Es necesaria también la instalación del Kit de Desarrollo o SDK de esta tecnología, pudiendo obtenerla de manera gratuita en http://www.silverlight.net/.

Este proyecto se ha desarrollado con la versión 5 del SDK, última versión existente al inicio del proyecto.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Con dicha instalación sólo tenemos acceso sino a un mínimo de herramientas básicas de desarrollo. Para obtener un conjunto más amplio de herramientas sin tener que crear estructuras de datos complicadas, debemos instalar los últimos toolkits.

Este complemento nos permite la inclusión en nuestra página web de selectores de fechas, cajas para la selección de números, gráficos con diseños autogenerados, etc.

Para finalizar citar también que se ha necesitó de un Servidor de Bases de Datos. Por maximizar la compatibilidad, y ya que el sistema de almacenamiento de datos [[1]] se desarrolló en ese servidor, se ha optado por un Microsoft SQL Server 2008.

4.2 FUNCIONAMIENTO GENERAL

4.2.1 ESTRUCTURA DE LA APLICACIÓN SILVERLIGHT

El desarrollo es una página web desarrollada con Microsoft Silverlight, como en algún apartado anterior se ha comentado. La web se compondrá de una página principal que contendrá a las demás páginas web.

El contenedor principal (equivalente a una "Master Page" de las tecnologías .NET) es una página web que hereda de la clase MainPage. Se trata de una clase que tiene su propio contenido visual y navegación y se verá como una parte fija en toda la aplicación. Además, esta clase contendrá variables y datos globales de toda la aplicación, al igual que algún método que serán usados en todas las páginas.

Los datos almacenados en esta página principal son:

- Datos de usuario: Identificación del usuario con el que se ha accedido al sistema, su nombre y el idioma predeterminado de acceso.
- Datos de Rol: Identificador y permisos del Rol de acceso (Administrador, Datos o Consulta) con el que el usuario ha accedido al sistema.
- Estado Actual: Clase VisualState, que representa los distintos tipos de páginas que pueden ser visualizadas. Hay un estado para Administradores, otro para Datos y otro para Consulta, así como un estado inicial para quien no se ha autentificado que muestra la pantalla de acceso.
- Listado con todos los Roles disponibles para el usuario actual. Se almacena una vez que el usuario se ha autentificado ya que para el acceso se le debe presentar la opción de usar uno de ellos. Una vez que se ha obtenido, se mantiene para mayor rapidez en futuras consultas.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Los métodos que implementa permiten distintas acciones desde cualquier página así se evita duplicidad de código:

- Cargar una página en el contenedor de la página principal. Así se facilita la navegación para los enlaces presentes en todas las páginas.
- Salir del programa, cerrando la sesión y cambiando al estado visual de no autentificados.
- Cambiar el idioma predeterminado de un usuario en la base de datos.
- Acceder a las funcionalidades de importar una factura desde XML y comprobar una factura, módulos desarrollados en otro PFC [[1]], pudiendo cargar automáticamente las facturas proporcionadas por Endesa.
- Acceder a las funcionalidades de importar una factura desde un fichero Excel, pudiendo cargar automáticamente las facturas proporcionadas por Unión Fenosa.

Además la aplicación Silverlight está incluida en una página HTML como un objeto de la aplicación para poder ser servido por cualquier servidor de páginas web. Este documento HTML será el que ponga las cabeceras y pies de página necesarias para seguir el manual de estilo del SNS-O.

Este es el documento HTML, contenedor de las páginas realizadas en Silverlight:

```
<div id="header">
    <div id="navarraes" valign="center">
        <a href="http://www.navarra.es">
            <img align="left" src="Images/logo.navarra.es.gif" alt="navarra.es"></img>
        </a>
    </div>

    <div id="snso">
        <a  href="http://www.navarra.es/home_es/Gobierno+de+Navarra/Organigrama/Los+departamentos/Salud/Organigrama/Estr
            <img align="right" src="Images/snso2.jpg" alt="sns-o"> </img>
        </a>
    </div>
</div>


<div id="silverlight" >
    <object id="silverlight" width="60em" height="620"
        data="data:application/x-silverlight-2,"
        type="application/x-silverlight-2" >
        <param name="source" value="ClientBin/SNSO.xap"/>
    </object>
</div>

<div id="bottom">
    <div id="gna">
        <a href="http://www.navarra.es/">
            <img align="left" src="Images/gobierno.jpg" alt="Gobierno de Navarra"></img>
        </a>
    </div>

    <div id="bottomtext">
        <a href="http://www.navarra.es/home_es/Indices/Sugerencias/default.htm" id="bottomlink">Contacto / Kontaktua</a>
    </div>
</div>
iv>
```

Figura 66: Código HTML de la Página Contenedora

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Para facilitar comprensión de toda la estructura del sistema, se puede observar este esquema que representa la estructuración:

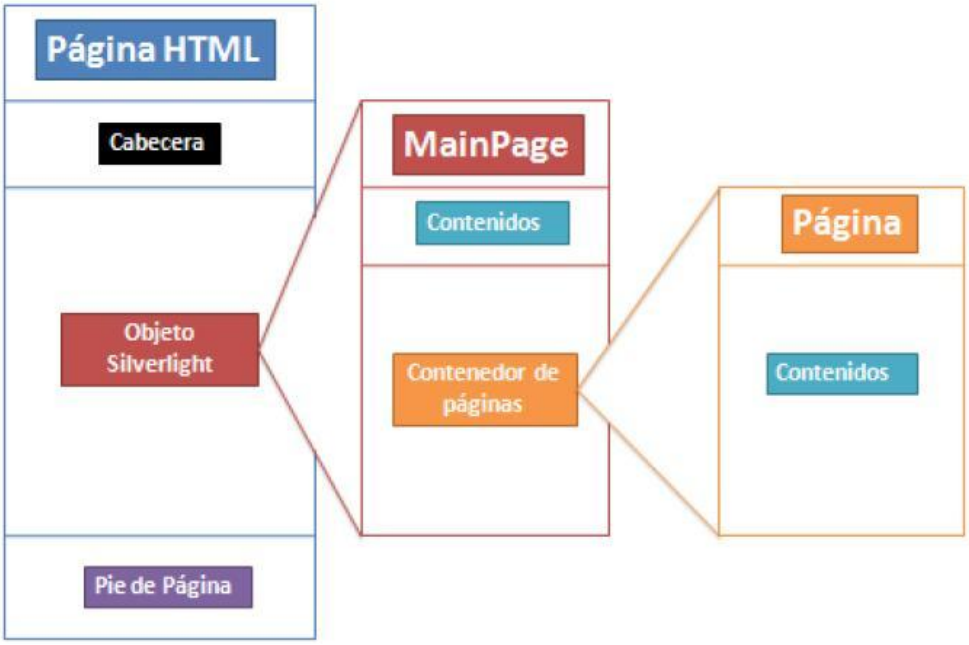


Figura 67: Esquema de Páginas

Y el aspecto general de la aplicación, una vez construidos todos los contenedores es este:

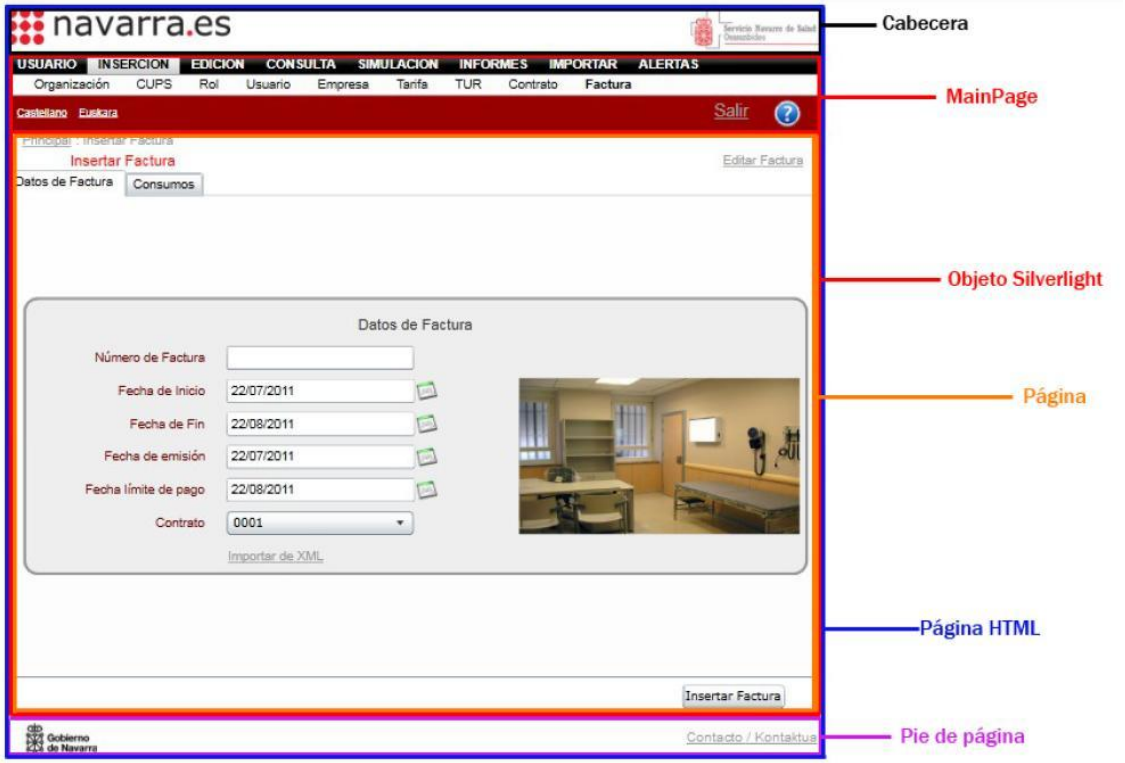


Figura 68: Estructura de la Aplicación

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.2.2 NAVEGACIÓN

Como ya se ha mostrado, las páginas de contenidos son mostradas en un contenedor de la página principal.

Para el usuario, estas páginas del contenedor son lo único que cambia. Para efectuar la navegación existen distintos métodos.

Como indican las normas de estilo, debe existir una barra de navegación primaria que contenga las categorías a las que podemos acceder, y que al pulsarla se muestren debajo de ella las páginas a las que podemos navegar o las acciones que se puedan efectuar. Además, existen otros enlaces que generan la navegación entre las páginas.

Figura 69: Barra de Navegación

Estos botones están desarrollados utilizando bloques de texto y no simples enlaces, ya que no queremos una navegación plana. El sistema para saber a qué página se debe acceder es el que se indica a continuación:

El nombre del campo de texto que se quiera utilizar está compuesto por dos partes, el nombre de la página a la que se quiera acceder y un posible número, ya que puede ser que en la misma página haya dos enlaces al mismo lugar, siendo imposible tener dos campos de texto con el mismo nombre.

El sistema de navegación obtiene el nombre del campo de texto en el que hayamos clicado, le quita el número si lo hubiera y navega a la página correspondiente.

```
//Al hacer click se pone en negrita
private void SecondaryBarClick(object sender, MouseButtonEventArgs e)
{
    TextBlock senderBlock = (TextBlock)sender;
    string[] nameVector;
    string name;
    Regex ex = new Regex("[0-9]"); ;

    //Primero debemos desactivar el que ahora mismo está activado
    deactivateSecondaryBarCurrentButton();

    selectSecondaryButton(senderBlock);

    /*Navigation. En general para simplificar las páginas a las que podemos acceder
    * se llamarán como el botón de la barra secundaria con el que se accede a ellas. Puede
    * que tengan un número al final, ya que hay varias barras con los mismos botones y no pueden
    * tener el mismo nombre*/

    //Cogemos el nombre, quitándole el número del final
    nameVector = ex.Split(senderBlock.Name);
    if (nameVector.Length > 0)
    {
        name = nameVector[0];
        loadPage(name); //Cargamos la página
    }
}
```

Figura 70: Clic en Enlace

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El uso de campos de texto y este sistema de navegación concreto es porque se pretende mantener las páginas almacenadas por dos motivos: primero, mantener el estado de estas, con los posibles datos que ya hubiera introducido el usuario y segundo, agilizar la carga de aquellas páginas que ya se hayan visitado, almacenándolas en modo local y no teniendo que volver a pedirlas al navegador.

Para almacenar las páginas se utiliza una tabla hash de páginas (clase Page) en la que se almacenan en el momento que se cargan por primera vez. Cuando se le indica a la página principal que debe navegar a una página, la primera acción que hace es buscar en la tabla. Si encuentra la página en la tabla es que ya ha sido cargada anteriormente y la coloca en el contenedor. Si la página no está, navega hacia ella. Como en los nombres de los enlaces no se hace distinción del idioma sobre el que queremos cargar la página, el propio sistema de navegación comprueba el idioma actual y si es euskara añade una E al final del nombre, ya que todas las páginas en euskara tienen el mismo nombre que las que están en castellano, con una letra E al final. Sin embargo, como más tarde hablaremos, la aplicación sólo se ha desarrollado en un idioma, en castellano.

```
//Cargar una página
public void loadPage(string pageName)
{
    string relativeName;
    Page page;

    //Dependerá del idioma en el que estemos, excepto en la página de Login
    if(pageName.Equals("Login"))
        relativeName="/"+pageName;
    else if (user.idioma.Equals("Euskara"))
        relativeName = "/EU/" + pageName +"E";
    else
        relativeName = "/ES/" + pageName;

    //Si la página ha sido cargada anteriormente, la usamos
    if (chargedPages.TryGetValue(pageName, out page))
        ContentFrame.Content = page;

    else //Si la página no está cargada navegamos. Almacenaremos la página una vez cargada
        ContentFrame.Navigate(new Uri(relativeName, UriKind.Relative)).ToString();
}
```

Figura 71: Navegación

4.2.3 INTERACCIÓN CON EL USUARIO

Cuando el usuario pide que se ejecute alguna acción, el sistema interactúa con él para proporcionarle los datos que necesita sobre la acción que ha solicitado. Para ello el sistema genera una serie de ventanas *pop-up* que mostrarán al usuario todos los datos necesarios.

Las ventanas que se muestran pueden ser de tres tipos diferentes, dependiendo del momento y de la acción solicitada por el usuario:

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- Ventana de Error: Si alguno de los datos introducidos por el usuario no tiene el formato correcto (por ejemplo si debe indicar un valor numérico y ha indicado texto), si en una inserción ya existe el ítem a insertar o si ha sucedido cualquier otro tipo de problema, el sistema muestra una ventana de error para avisar al usuario de esta circunstancia, y de que la acción que intentaba realizar no se ha completado.

Figura 72: Ventana de Error

- Ventana de Confirmación: Si el usuario ha pulsado un botón que implica que un objeto será insertado, eliminado o modificado en el sistema, con las consiguientes consultas SQL de por medio, antes de ejecutarse la acción se le dará siempre la opción de comprobar lo que quiere hacer (datos que va a introducir, modificaciones que va a hacer, etc.), mostrando en la ventana un resumen de todos los datos, y podrá ejecutar la acción o cancelarla.

Figura 73: Ventana de Confirmación

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- Ventana de selección: En algunos casos, el usuario deberá seleccionar un objeto, por ejemplo cuando decida modificar datos que ya existan en el sistema. Para efectuar la selección se utiliza una ventana que contiene un listado de todos los elementos disponibles.

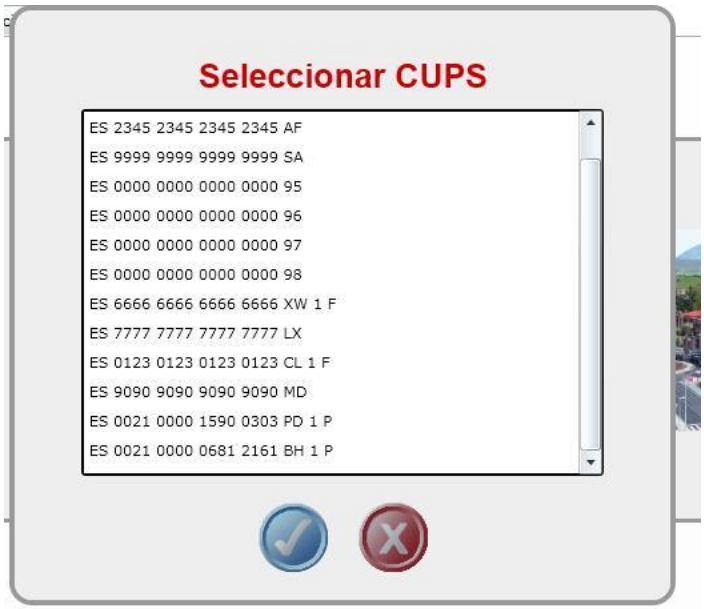


Figura 74: Ventana de Selección

4.2.4 INSERCIÓN DE DATOS

Cuando el usuario quiere introducir un nuevo dato en el sistema, primero accede a la página indicada (si es que tiene los permisos necesarios), rellena todos los campos obligatorios y pulsa el botón de insertar. Puede que algunos de estos campos a rellenar contengan información obtenida de la Base de Datos, como las listas desplegables o campos autocompletables.

Si se da este caso, justo en el momento de cargar la página y rellenar el campo se realizará la consulta contra la base de datos, obteniendo así toda la información necesaria para realizar la inserción. Ocurre lo mismo en las ediciones.

Una vez que el usuario ha completado el formulario rellenando todos los datos, pulsa el botón de inserción y se comprueba si todos los datos tienen el formato correcto (números, fechas…) y en el caso de ser así aparece la ventana de confirmación para que el usuario compruebe los datos a insertar. Si el usuario queda conforme y pulsa aceptar, se envían los datos y se produce la instrucción INSERT. En el caso de existir algún error un pop-up aparecerá tal y como se ha descrito en el anterior apartado.

Además, en todas las páginas de inserción de datos encontramos un enlace en la parte superior derecha de la página desde el que acceder a la página de edición del mismo elemento. Esto facilita las cosas a la hora de reparar pequeños errores tipográficos que puedan producirse a la hora de insertar, teniendo un acceso rápido.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.2.5 MODIFICACIÓN DE DATOS

Cuando un usuario desea cambiar algún dato, todos sus campos pueden ser modificados (excepto el identificador autonumérico para la Base de Datos, que en ningún momento es visualizado por el usuario). En cualquier caso, todos los ítems tienen, desde el punto de vista del usuario un identificador textual (nombre, descripción, número…) que debe ser único, pero que también debe poder cambiarse.

Por tanto, cuando un usuario quiere cambiar un dato lo primero que debe hacer es seleccionarlo a través de la ventana que se generará, tal y como en el apartado anterior se ha comentado. El objeto seleccionado se queda almacenado hasta que se termina todo el proceso, para poder mantener de esta manera todos los identificadores originales.

Al igual que en las inserciones, cuando todos los datos han sido introducidos y son correctos, la información es enviada a la base de datos para efectuar los comandos SQL correspondientes. Hay que recordar que el sistema mantiene históricos de todos los elementos que hayan sucedido en un determinado lugar, para una determinada tarifa…

Es por esto que aunque en algunos casos los objetos de la base de datos son modificados (cambios por errores, equivocaciones…) en la mayoría de los casos se insertan nuevos datos, almacenando los periodos de tiempo en los que un objeto mantuvo unos determinados valores.

4.2.6 BORRADO DE DATOS

Al igual que en el caso anterior no se modificaban los datos (para mantener los históricos y la integridad de la información), los borrados no son borrados como tales sino que se indica en la base de datos que han sido dados de baja.

Entonces, aunque desde el punto de vista de un usuario el elemento ha sido eliminado y desaparece del sistema, la instrucción SQL ejecutada es un UPDATE, en el que se cambia la fecha de fin del elemento, indicando así que ya no está activo, y así se mantendrá toda la información que se introduzca en el sistema, sea la vigente o no.

4.2.7 ASOCIACIONES Y SELECTORES

En muchos casos el usuario debe realizar asociaciones lógicas que el programa traducirá como asociaciones entre tablas de la Base de Datos. Asociará CUPS a una organización, CUPS a un Rol, organizaciones entre sí… Para ello se ha generado un selector que permite navegar por las jerarquías organizativas, los CUPS, los roles… dependiendo de la página.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El funcionamiento es muy parecido en todos los casos: Utilizando los botones, las teclas o haciendo clic en los elementos se navega y se muestran los datos. Cuando estos se han llevado a la lista de asociación, y se efectúa la inserción o modificación, las asociaciones son generadas. Desde el punto de vista del usuario, los elementos que se hayan quedado en la última lista será los que se mantengan, y el resto no tendrán ninguna asociación. En las modificaciones, para el usuario las asociaciones que antes estuvieran y ahora no, son eliminadas.

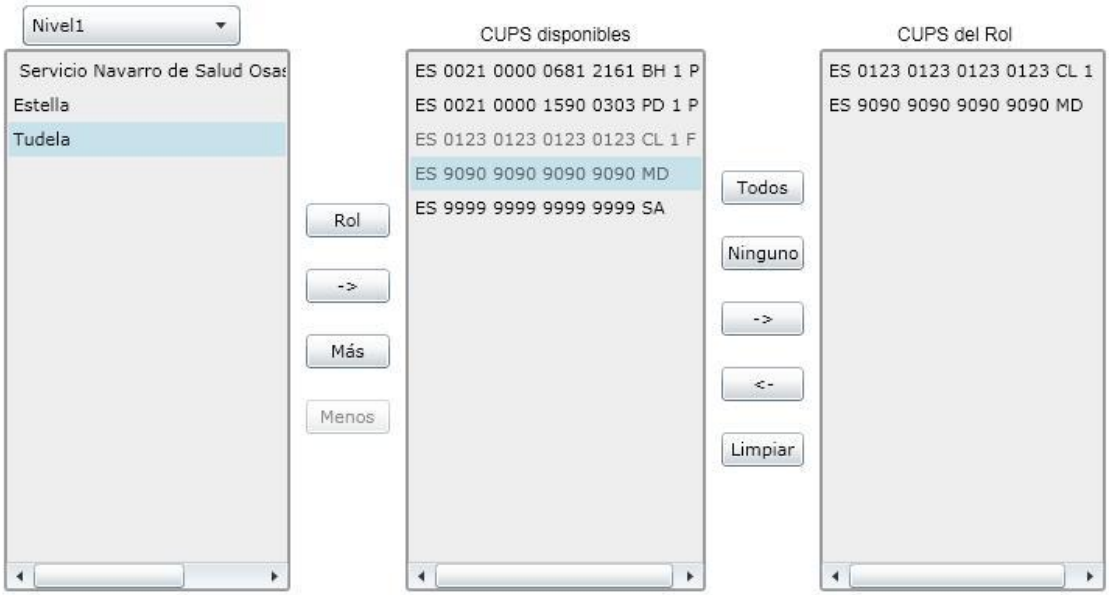


Figura 75: Selector para Asociaciones

En cualquier caso, como ya se ha repetido en varias ocasiones anteriormente, en la base de datos las modificaciones se deben mantener, indicando las fechas de inicio y fin de cada asociación, para que se mantenga la integridad y se puedan obtener históricos.

Por tanto, aunque al insertar un nuevo elemento el hecho de crear las asociaciones sí que sea tan sencillo como efectuar instrucciones INSERT sobre la base de datos, no es así en el caso de que se esté modificando un elemento, en el caso de que la asociación sea de muchos a muchos.

En estas asociaciones múltiples, se entiende sobre la base de datos que una asociación está activa si su fecha de fin contiene el valor NULL. En cambio, una asociación cuya fecha de fin contenga una fecha válida, significa que la asociación ha terminado y no es la vigente.

Es por todo esto que en las modificaciones se debe encontrar un sistema más complicado, que se encargue de comprobar si una asociación es nueva (y si es así efectuar las instrucciones INSERT necesarias), si es una asociación que ya mantenía (en cuyo caso no haremos ningún cambio) o si por el contrario se debe eliminar alguna asociación existente (en ese caso pondremos un valor en la fecha de fin de la relación).

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Para implementar todo esto se han seguido los siguientes pasos, que se muestran ejemplificados con fragmentos de código asociados a la edición de una organización de Nivel 4 y sus asociaciones con CUPS.

- Se genera una tabla auxiliar, con un nombre aleatorio y con el mínimo posible de probabilidades de ser repetido, concatenando para su nombre la palabra *aux*, el nombre de la tabla sobre la que se harán las modificaciones, el identificador del elemento que estamos modificando y el número de milisegundos actuales.

```
//Reasociamos los CUPS
//Usaremos una tabla auxiliar
string tableName = "Aux_Asoc4_CUPS_" + level4Id + "_" + DateTime.Now.Millisecond;

create = "CREATE TABLE " + tableName + "(" +
         "[idCUPS] [int] NOT NULL, " +
         "[idNivel4] [int] NOT NULL);";

dc.ExecuteCommand(create);
```

Figura 76: Creación de Tabla Auxiliar

- Se encuentran almacenado en una lista todos y cada uno de los elementos que el usuario ha dejado seleccionados en la lista final. Se insertará en esta nueva tabla auxiliar cada uno de ellos

```
//Para cada CUPS a insertar
foreach (int CUPSId in CUPSList)
{
    //Insertamos
    insert = "INSERT INTO " + tableName + " " +
           "VALUES(" + CUPSId + "," + level4Id + ");";

    dc.ExecuteCommand(insert);

}
```

Figura 77: Inserción de Tabla Auxiliar

- Se introduce en la tabla original cada uno de los nuevos elementos**,** es decir, aquellos que no tuviesen una asociación activa. Para obtener cuáles son dichos nuevos elementos se efectúa una operación de diferencia: los obtenemos al sustraer de la tabla auxiliar los elementos que están insertados (y activos) en la tabla original.

```
//Insertamos en la tabla real los nuevos
insert = "INSERT INTO Asoc4_CUPS " +
             "SELECT idCUPS,idNivel4,GETDATE(),NULL " +
                "FROM " + tableName + " " +
                "WHERE idCUPS IN( " +
                    "SELECT idCUPS " +
                    "FROM " + tableName + " " +
                    "WHERE idNivel4=" + level4Id + " " +
                    "EXCEPT  " +
                    "SELECT idCUPS " +
                    "FROM Asoc4_CUPS " +
                    "WHERE idNivel4=" + level4Id + " " +
                ") AND idNivel4=" + level4Id + ";";
```

Figura 78: Inserción en la Tabla Original

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- Se modifican las fechas de fin de asociación en la tabla original para aquellos elementos que ya no deban estar asociados. Para ello se hace la misma operación que en el caso anterior, invirtiendo el orden de los factores: Obtenemos los elementos a eliminar quitando a los elementos de la tabla original los elementos de la tabla nueva.

```
//Damos de baja en la tabla real los correspondientes
delete = "UPDATE Asoc4_CUPS SET FechaFin_Asoc4C=GETDATE()" +
         "WHERE idCUPS IN(" +
                "SELECT idCUPS " +
                "FROM Asoc4_CUPS " +
                "WHERE idNivel4=" + level4Id + " " +
                "EXCEPT  " +
                "SELECT idCUPS " +
                "FROM " + tableName + " " +
                "WHERE idNivel4=" + level4Id + " " +
        ") AND idNivel4=" + level4Id + ";";

dc.ExecuteCommand(delete);
```

Figura 79: Dar de Baja en la Tabla Original

- Por último se elimina la tabla auxiliar.

```
//Eliminamos la tabla auxiliar
drop = "DROP TABLE " + tableName + ";";
dc.ExecuteCommand(drop);
```

Figura 80: Borrado de la Tabla Auxiliar

4.2.8 FACILIDAD DE OPERACIÓN: TRABAJO CON EL TECLADO

Para facilitar el trabajo con la herramienta, se ofrece la posibilidad de ejecutar todas las acciones dentro de una misma página sin necesidad de emplear el ratón. Las acciones que se pueden ejecutar son las siguientes:

- Insertar un elemento pulsando la tecla *enter*, de la misma manera que el botón de inserción.
- Modificar un elemento pulsando la tecla *enter*, de la misma manera que el botón de edición.
- Navegación por todos los campos de datos de la manera habitual, mediante sucesivas pulsaciones de la tecla *tabulador.*
- Cierre de la ventana de error con las teclas *enter* y *escape*.
- En la ventana de selección de elemento, con la tecla *enter* se selecciona de la misma manera que con el botón de *ok* o efectuando un doble clic sobre el elemento a seleccionar.
- Cierre de la ventana de selección sin seleccionar ningún elemento con la tecla *escape.*

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- En la ventana de confirmación de inserción, edición o borrado, aceptación de los datos del mismo modo que con el botón de *ok.*
- Cierre de la ventana de confirmación de inserción, edición o borrado sin efectuar ningún cambio en la base de datos con la tecla *escape.*
- En la página de inserción de niveles, navegación entre los distintos niveles con las teclas *Avance de Página* y *Retroceso de Página* para ir a los niveles inferiores y superiores respectivamente, generando la misma animación de navegación (cierre del cuadro de datos del nivel actual y apertura del indicado) que la selección con el ratón de un nivel específico.

4.3 CONEXIÓN CON LA BASE DE DATOS

4.3.1 SILVERLIGHT

Microsoft Silverlight está diseñado para ser, ante todo, lo más seguro posible. Por ello no nos permite crear una conexión directa con la Base de Datos (todo lo contrario que el resto de tecnologías .NET).

Figura 81: Conexión con Base de Datos de Microsoft Silverlight

Microsoft Silverlight posee un sistema con servicios web. La aplicación se conecta de manera asíncrona con diferentes servicios web, que son los encargados realmente de conectarse con la base de datos y ejecutar las consultas. En caso de que la consulta

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

devuelva algún listado, el servicio web lo devuelve para que la aplicación pueda usarlo de la manera más conveniente.

4.3.2 CREACIÓN DE UN MODELO DE DATOS

El IDE Visual Studio nos facilita poder realizar conexiones con servidores de Microsoft SQL de una manera automática. Lo único necesario es generar un modelo de datos a partir de una conexión al servidor.

Visual Studio muestra un listado con todas las conexiones que podemos obtener. Seleccionando la indicada accedemos a la base de datos.

Una vez seleccionada la base de datos, crear el modelo de datos es tan sencillo como arrastrar las tablas y procedimientos almacenados desde el explorador de servidores.

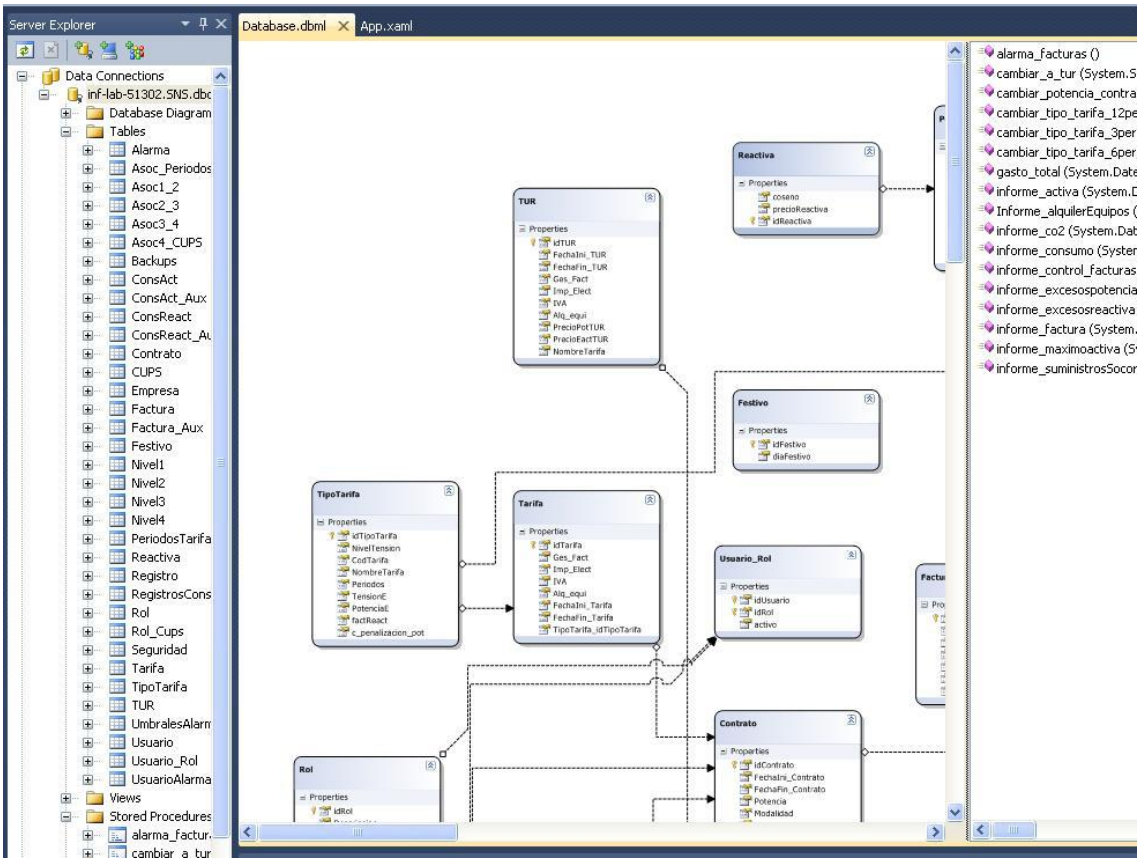


Figura 82: Creación del Modelo de Datos

Una vez generado el modelo de datos se creará una clase de Contexto de Datos de la base de datos. Es la encargada de abrir y cerrar las conexiones necesarias en los servicios web que implementemos y mediante ella podremos ejecutar todas las consultas que sean necesarias.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.3.3 CREACIÓN DE UN SERVICIO WEB

En la creación de un servicio web, hay que tener en cuenta que una clase de contexto de datos no devolverá una lista genérica de resultados, sino que almacenará en objetos los datos que obtenga, devolviendo un listado de dichos objetos.

Por lo tanto, antes de la creación de un servicio web en sí, es necesaria la creación de los objetos que el servicio web utilice o devuelva. Hay que tener en cuenta el modelo de datos ya ha generado de manera automática aquellas clases que representan objetos de cada una de las tablas introducidas en el diseñador del modelo, por lo que si la consulta únicamente devuelve resultados de una tabla, no será necesaria la creación de ninguna nueva clase.

Por desgracia, en este proyecto la gran mayoría de las consultas se obtienen a partir de la unión de varias tablas, por lo que se debe crear una clase específica para cada una de las consultas.

Estas clases deberán seguir un modelo muy estricto para que la consulta funcione sin problemas. Serán clases con únicamente los atributos (definidos como públicos), sin ningún método ni constructor. Tomarán el constructor por defecto. Si no se definen exactamente con el mismo nombre y tipo que la consulta que devuelva el servicio web existirán problema.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace SNSO.Web.Data
{
    public class CompanyObject
    {
        /* El servicio, al exportar esta clase reordenará los atributos por orden
         * alfabético, por tanto le damos nombres distintos y luego renombraremos.
           Además deben llamarse del mismo modo que el resultado de la BD*/
        public string atribA { get; set; } //Nombre
        public string atribB { get; set; } //CIF
        public string atribC { get; set; } //CNAE
        public string atribD { get; set; } //RazonSocial
        public string atribE { get; set; } //DirFiscal
        public string atribF { get; set; } //Dirección
        public string atribG { get; set; } //Teléfono
        public string atribH { get; set; } //Email
        public string atribI { get; set; } //Fax
        public string atribJ { get; set; } //PersonaContacto
        public string atribK { get; set; } //Móvil
    }
}
```

Figura 83: Clase para consulta de Empresa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Una vez generada la clase, ya se puede construir el servicio. En él servicio web se pueden implementar tantas consultas como se necesite, creando un método para cada una y marcándola con el atributo [OperationContract] delante de cada método.

En el método se puede ejecutar el código SQL a través del Contexto de Datos. Los

```
//Seleccionar todas las empresas
[OperationContract]
public List<CompanyObject> selectCompany()
{
    IEnumerable<CompanyObject> result;
    DatabaseDataContext dc = new DatabaseDataContext();

    string query = "SELECT DISTINCT Nombre atribA, CIF atribB, CNAE atribC, RazonSocial atribD, DirFiscal atribE, Direccion atribF," +
                "Telefono atribG, Email atribH,Fax atribI, PersonaContacto atribJ, Movil atribK " +
                "FROM Empresa " +
                "ORDER BY Nombre;";

    result = dc.ExecuteQuery<CompanyObject>(query,1,"Iberdrola");
    return result.ToList<CompanyObject>();
}
```

Figura 84: Consulta de Empresa

métodos devuelven listas de objetos que serán usadas para la generación de listados.

Por otro lado, si las consultas SQL no son para consultar datos sino para insertar datos, modificar o borrar, no se necesita la creación de ninguna clase.

```
[OperationContract]
//Insertar una Empresa. Devuelve 0 si se ha insertado correctamente, 1 si es un CIF repetido, si es 2 es un Nombre repetido. 3 Error de
public int insertCompany(string CIF, string socialReason,string fiscalAddress, string CNAE, string name,string address,string email,
    string phone,string fax, string contact, string mobile)
{
    DatabaseDataContext dc = new DatabaseDataContext();

    //Iniciar transacción
    dc.Connection.Open();
    dc.Transaction = dc.Connection.BeginTransaction();


    string insert = "INSERT INTO Empresa(CIF,RazonSocial,DirFiscal,CNAE,Nombre,Direccion,Email,Telefono,Fax,PersonaContacto,Movil) " +
                    "VALUES('" + CIF +
                    "','" + socialReason +
                    "','" + fiscalAddress +
                    "','" + CNAE +
                    "','" + name +
                    "','" + address +
                    "','" + email +
                    "','" + phone +
                    "','" + fax +
                    "','" + contact +
                    "','" +mobile + "');";

    try
    {
        //Ejecutar
        dc.ExecuteCommand(insert);
        //Completar transacción
        dc.SubmitChanges();
        dc.Transaction.Commit();
        return 0;
    }
    catch (SqlException ex)
    {
        //Cancelar la transacción
        dc.Transaction.Rollback();

        if (ex.Number == 2627) //Error de restricción
        {
            if (ex.Message.Contains("'empresa_cons'")) //Restricción de CIF
                return 1;
            else if (ex.Message.Contains("'cons_empl_nom'")) //Restricción de nombre
                return 2;
            else
                return 3;
        }
        else return 3; //Error desconocido
    }
}
```

Figura 85: Consulta de Inserción deEmpresa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

En estos casos sólo se deberá ejecutar el comando con los parámetros necesarios, utilizando el contexto de datos de la misma manera.

Además, en estas últimas consultas que se introducen o modifican datos se devuelve un valor para indicar el resultado del comando. Así, si la transacción ha tenido éxito se devolverá un valor, pero existirán otros posibles valores que indiquen errores de cualquier otro tipo, como por ejemplo el intento de inserción de un elemento UNIQUE ya existente, o si no se conoce la causa del error, se devuelve un número indicando que es un error desconocido.

Dichos errores se darán como excepciones de la clase SqlException, por lo que para poder devolver el error correcto, se deberá capturar la excepción y analizar su valor.

4.3.4 INYECCIÓN SQL

En algunos casos se realizan consultas usando datos introducidos por el usuario, por ello hay que tener en cuenta que existe la posibilidad de que inserte código malicioso. Para evitar la inyección SQL, las funciones de ejecución ya dan la posibilidad de usar parámetros.

Cuando el servicio ejecuta una consulta con parámetros, aquellos parámetros que son introducidos entre llaves con un valor numérico, antes de reemplazar los valores genera las secuencias de escape necesarias para evitar código malicioso.

```
[OperationContract]
/* Insertar una factura. Devuelve 0 si se ha insertado correctamente, 1 si es número de factura repetido y 3 error desconocido*/
public int insertInvoice(DateTime initDate, DateTime endDate, string number, DateTime payDate, int contractId, DateTime emissionDate,
    List<Consume> consumeList, int userId)
{
    //Sólo se quiere la fecha, no la fecha y la hora
    initDate = initDate.Date;
    endDate = endDate.Date;
    payDate = payDate.Date;
    emissionDate = emissionDate.Date;

    DatabaseDataContext dc = new DatabaseDataContext();

    //Iniciar transacción
    dc.Connection.Open();
    dc.Transaction = dc.Connection.BeginTransaction();

    int invoiceId;
    string query;

    string insert = "INSERT INTO Factura(FechaIni_Factura,FechaFin_factura,NumFactura,FechaLimitePago,Contrato_idContrato,FechaEmision) " +
                    "VALUES({0},{1},{2},{3},{4},{5});";

    try
    {
        dc.ExecuteCommand(insert, initDate, endDate, number, payDate, contractId, emissionDate);
```

Parámetros numéricos entre llaves

Figura 86: Ejecución de Consulta con Parámetros

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.3.5 TRANSACCIONES

En muchas ocasiones los métodos que se realizan inserciones o modificaciones no ejecutan una única instrucción o comando SQL sino que se ejecutan varias y son secuenciales. El que no se ejecute correctamente una de ellas afecta a la ejecución de las siguientes o la inserción de unas sí y otras no en la base de datos provocaría inconsistencias, por ello se usan transacciones.

Al inicio del método que necesite usarla, se abre una transacción, y si todas las operaciones se han ejecutado correctamente se procederá al volcado de todos los datos sobre la base de datos, de lo contrario la transacción será cancelada y no se producirá ningún cambio en ésta, manteniéndose intacta.

```
[OperationContract]
//Insertar una Empresa. Devuelve 0 si se ha insertado correctamente, 1 si es un CIF repetido, si es 2 es un Nombre repetido. 3 Error de
public int insertCompany(string CIF, string socialReason,string fiscalAddress, string CNAE, string name,string address,string email,
    string phone,string fax, string contact, string mobile)
{
    DatabaseDataContext dc = new DatabaseDataContext();

    //Iniciar transacción
    dc.Connection.Open();
    dc.Transaction = dc.Connection.BeginTransaction();
```

Abertura de la transacción

Código de las inserciones o modificaciones necesarias

```
    try
    {
```

Ejecución de los códigos

```
        //Completar transacción
        dc.SubmitChanges();
        dc.Transaction.Commit();
        return 0;
    }
    catch (SqlException ex)
    {
        //Cancelar la transacción
        dc.Transaction.Rollback();

        if (ex.Number == 2627) //Error de restricción
        {
            if (ex.Message.Contains("'empresa_cons'")) //Restricción de CIF
                return 1;
            else if (ex.Message.Contains("'cons_empl_nom'")) //Restricción de nombre
                return 2;
            else
                return 3;
        }
        else return 3; //Error desconocido
    }
}
```

Volcado de los datos

Cancelación de la transacción

Figura 87: Estructura de una Transacción

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.3.6 LLAMADA AL SERVICIO WEB

Las llamadas a los servicios web se efectúan de manera asíncrona. En primer lugar se genera la llamada al método correspondiente del servicio web indicado. Éste iniciará la ejecución de forma inmediata y cuando termine, si tiene algún dato para devolver, entonces enviará un aviso al cliente.

Por ello el cliente deberá tener definido un método que será llamado cuando el servicio haya terminado de procesar los datos, ejecutándose entonces el código necesario, para mostrar un listado o en caso de error mostrar un aviso.

```
//Llamar a la consulta de CUPS
public void callSelectCupsQuery(bool notActive,bool notLinked,RolAsoc role,string date)
{
    //Activamos el indicador de espera
    indicator.IsBusy = true;

    //Servicio
    DatabaseServiceClient client = new DatabaseServiceClient();

    //Al ser una llamada asíncrona, al acabar iremos a otra función
    client.selectCUPSQueryCompleted += new EventHandler<selectCUPSQueryCompletedEventArgs>(CUPSQueryCompleted);
    client.selectCUPSQueryAsync(notActive,notLinked,role.Permiso,role.idRol,date); //Llamamos de manera asíncrona
}

//Consulta finalizada
void CUPSQueryCompleted(object sender, selectCUPSQueryCompletedEventArgs e)
{
    grid.ItemsSource = e.Result;

    //Desactivamos el indicador de espera
    indicator.IsBusy = false;
}
```

Figura 88: Llamada al Servicio Web

Al ser un proceso bastante engorroso, ya que no se efectúa de manera secuencial, en este proyecto se optó por la creación de unos conectores auxiliares con la Base de Datos, encargados de llamar a los servicios y rellenar el listado correspondiente.

Para ello el conector tendrá como atributos un apuntador al listado sobre el que se mostrarán los datos, así como a un indicador del estado de la actividad. Al iniciar la consulta pondrá el indicador de actividad en espera (provocando que el usuario visualice un icono que algo se está procesando en la pantalla) y en el método de devolución de datos lo desactivará y rellenará el listado, como se puede observar en la figura 85.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
using System;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Ink;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using SNSO.DatabaseQueryServiceClient;
using SNSO.Data.Objects;
using SNSO.LoginUtilityServiceClient;

namespace SNSO.Data
{
    //Clase auxiliar para la conexión con la base de datos
    public class DatabaseGridQueryConnector
    {
        //Elementos sobre los que mostramos los datos
        private DataGrid grid;
        private BusyIndicator indicator;

        //Constructor
        public DatabaseGridQueryConnector(DataGrid grid,BusyIndicator indicator)
        {
            this.grid = grid;
            this.indicator = indicator;
        }
```

Figura 89: Declaración y Constructor de un Conector

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.3.7 PUBLICACIÓN DE LOS SERVICIOS WEB

Para poner en funcionamiento los Servicios Web desarrollados, se debe tener instalado el complemento de Windows (Panel de Control, Programas y Características, Activar o Desactivar las Características de Windows) Internet Information Services (IIS), con todos los módulos de publicación de servicios web, el cuál será el servidor que sirva las páginas web.

La publicación la realiza de manera automática Visual Studio. Para ello se debe ir a la sección Web de las Preferencias del Proyecto, activar la casilla de utilización del servidor IIS local (en vez del servidor propio de Visual Studio) y generar los directorios requeridos, además de cambiar las rutas de los servicios Web, ya que se necesitan que estén en localhost.

Para que los servicios sean accesibles desde distintos puntos, se debe crear en el mismo directorio desde el que se están sirviendo un documento llamado clientaccesspolicy.xml (específico de Silverlight) o crossdomain.xml (creado primeramente para aplicaciones Flash, pero Silverlight también lo soporta). Con el primero de ellos es suficiente para nuestro caso. Deberá tener la siguiente configuración:

```
<?xml version="1.0" encoding="utf-8"?>
<access-policy>
  <cross-domain-access>
    <policy>
      <allow-from http-request-headers="SOAPAction">
        <domain uri="*"/>
      </allow-from>
      <grant-to>
        <resource path="/" include-subpaths="true"/>
      </grant-to>
    </policy>
  </cross-domain-access>
</access-policy>
```

Figura 90: Clientaccesspolicy.xml

En la sección allow-from se deberá indicar aquellos lugares desde los que está permitido el acceso a los servicios web. En el prototipo actual, al ser una aplicación de prueba se mantiene abierta la posibilidad de acceder desde cualquier URI, pero una vez puesto en funcionamiento, se limitará al servidor que sirva las páginas.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.4 ACCESO Y SEGURIDAD

Solamente los usuarios autorizados por un administrador podrán entrar al sistema. Cuando un usuario llega a la página de acceso, se le solicita su nombre de usuario y contraseña. Una vez introducida, el sistema envía una consulta al servidor de Bases de Datos, que devolverá el identificador del usuario y su idioma si existe y la contraseña (encriptado mediante funciones de SQL Server 2008) coinciden, usando consultas parametrizadas para evitar la inyección SQL.

```
[OperationContract]
//Seleccionar un usuario
public IEnumerable<LoginUser> getUser(string user, string password)
{
    IEnumerable<LoginUser> result;
    DatabaseDataContext dc = new DatabaseDataContext();

    string query = "SELECT idUsuario,usuario,idioma FROM Usuario " +
                   "WHERE usuario={0} " +
                   "AND PWDCOMPARE({1},contrasena)=1;";

    result = dc.ExecuteQuery<LoginUser>(query, user,password);
    return result.ToList<LoginUser>();
}
```

Figura 91: Consulta de Acceso

Una vez que la web recibe la respuesta, vuelve a comprobar que todo ha sido correcto, es decir, que la consulta ha devuelto algún valor y es un valor válido. En caso de que el acceso no haya sido correcto se mostrará al usuario el error.

Si el usuario y su contraseña son válidos, se procede a ejecutar una consulta para comprobar sus roles de acceso. Si el usuario posee más de un rol el usuario podrá elegir con qué rol entrar al sistema.

Figura 92: Acceso al Sistema

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Si por el contrario el usuario solo posee un único rol, éste entrará al sistema directamente sin aparecer el contenedor de roles.

4.4.1 ESTADOS

Todos los controles de una página Silverlight, incluida la propia página, pueden mantener distintos Estados Visuales, es decir, sus atributos tendrán unos valores determinados para cada estado, y mediante la clase VisualStateManager se puede alternar entre unos y otros.

También se pueden insertar animaciones para que sucedan al inicio de un estado, llevando los atributos de los valores actuales a los valores que tendrían que tener en el nuevo estado de forma animada.

En este caso los estados se han definido para la página principal, que es la que contendrá al resto de páginas, pudiendo comprobar así en todo momento sea cual sea la página por la que el usuario esté navegando cuál es el estado que mantenemos.

```
<!--Main Page States-->
<vsm:VisualStateManager.VisualStateGroups>
    <!--Grupo de estados-->
    <vsm:VisualStateGroup x:Name="StateGroup">
        <!--Sin animación-->
        <vsm:VisualStateGroup.Transitions>
            <vsm:VisualTransition GeneratedDuration="00:00:00"/>
        </vsm:VisualStateGroup.Transitions>

        <!--Estado por defecto: vacío-->
        <vsm:VisualState x:Name="BeginState"/>

        <!--Login-->
        <vsm:VisualState x:Name="LoginState">
            <Storyboard x:Name="ToLogin">
                <ObjectAnimationUsingKeyFrames Storyboard.TargetProperty="(UIElement.Visibility)" Storyboard.TargetName="Logout">
                    <DiscreteObjectKeyFrame KeyTime="0">
```

Figura 93: Definición del Grupo de Estados

Se definen cuatro estados para la página principal, a los que se accede con una animación sin tiempo de inicio, es decir, el usuario no percibe ninguna animación de cambio de estado, sino que se realiza el cambio de manera instantánea.

Estos son los estados definidos:

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- Estado de Login o estado previo al acceso: el estado en el que se encuentra la aplicación antes de que el usuario introduzca sus datos. Las barras de navegación e idiomas permanecen desactivadas. La única página a la que se puede acceder en este estado es la página de acceso.

Figura 94: Estado de Login

- Estado de Administrador: estado al que llega la aplicación si el usuario decide acceder con un Rol de Administrador. La barra de idiomas toma el color rojo y la barra de navegación muestra todas las opciones de la aplicación.

Figura 95: Estado de Administrador

- Estado de Datos: estado al que llega la aplicación si el usuario decide acceder con el Rol de Datos. La barra de idiomas toma el color verde y la barra de navegación muestra las opciones del usuario de Datos, es decir, alguna menos que el Rol de Administrador.

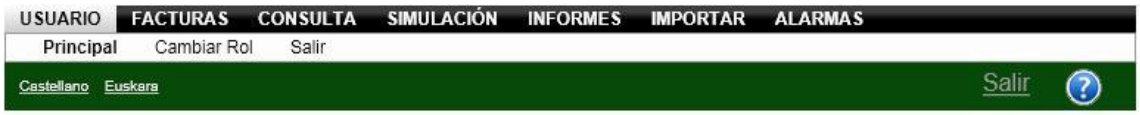


Figura 96: Estado de Datos

- Estado de Consulta: estado al que llega la aplicación si el usuario decide acceder con el Rol de Consulta. La barra de idiomas toma el color azul y la barra de navegación muestra únicamente las opciones de consulta: alguna menos que las del Rol de Datos.

Figura 97: Estado de Consulta

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

La razón por la cual se hace que la barra de idiomas cambie de color la de dar al usuario una referencia rápida de con qué Rol ha entrado a la aplicación y cuál está usando, sin tener que hacer ningún tipo de comprobación.

Cuando un usuario accede a una nueva página, en primer lugar el sistema comprueba su estado, solicitándoselo a la página contenedora. Tras obtener el nombre del estado como cadena de texto, comprueba si ese tipo de usuario tiene acceso a la página indicada. Si lo tiene, la navegación se efectuará de manera normal, en cualquier otro caso el estado cambiará al estado de acceso, volviendo a la página inicial como usuario desconectado. Así evitamos accesos indebidos a páginas restringidas.

```
// Executes when the user navigates to this page.
protected override void OnNavigatedTo(NavigationEventArgs e)
{
    //Comprobamos el estado. Sólo los administradores pueden insertar usuarios
    if (!main.currentState.Equals("AdminState"))
    {
        main.logout();
    }
```

Figura 98: Comprobación de Estados

Si el usuario, en la página principal de administración, datos y consulta dentro de la aplicación, solicita un cambio de estado, también es la página principal la que proporciona esta funcionalidad indicando al VisualStateManager qué elemento se quiere cambiar de estado (la propia página principal), el texto identificador del estado y una variable booleana para indicar si debe efectuar las animaciones, como se puede observar en la siguiente figura:

```
//Cambiar de estado
public void changeState(string state)
{
    VisualStateManager.GoToState(this,state,true);

    currentState = state;
}
```

Figura 99: Cambio de Estado

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.5 IDIOMA

Anteriormente la aplicación se implementó como sistema bilingüe, pudiendo alternar el castellano y el euskara.

Para la implementación del sistema bilingüe se barajó la idea de mantener las páginas en la Base de Datos, indexadas por idioma y nombre de página, solicitándolas al ser necesarias. Este sistema fue desechado, ya que al hacerse las conexiones a través de servicios web y ser una cantidad de información bastante amplia la que se debería intercambiar, el sistema podría ralentizarse de manera considerable.

Teniendo en cuenta que Silverlight carga las páginas en el momento en el que son solicitadas, y el sistema de almacenamiento de páginas en una tabla de Hash, se observó que la solución más sencilla y eficiente consistía en almacenar una página de cada idioma.

Así, mantiene un directorio /ES/ con las páginas en castellano, que será al que se accede al hacer la navegación desde la página principal, y de la misma manera, mantenemos también un directorio /EU/ con las páginas en euskara.

No obstante, el hecho de mantener en la misma aplicación páginas, que en el fondo son clases, con el mismo nombre creaba errores de compilación, por lo que se llegó a la solución de añadir al nombre de la página en euskara una letra E al final.

Las páginas son idénticas entre sí, tanto en implementación como en diseño, con la única diferencia de que cada texto que puede ser visto por el usuario ha sido traducido.

Por otra parte, la página principal sólo puede ser una, no pudiendo mantener una para cada idioma. La solución más sencilla consiste en crear un método que cambia de idioma cada uno de los atributos con texto visualizado por el usuario, que será llamada cuando la página se cargue o el usuario decida cambiar de idioma.

Un usuario siempre tendrá un idioma predeterminado, que se almacena en la base de datos, y que al acceder se comprueba para mostrarle unas u otras páginas. En todo momento podía cambiar de idioma haciendo clic en los botones de la parte superior izquierda (en la barra de color). Cuando lo hacía se le presentaba una pantalla de confirmación que le pregunta si quiere que ese idioma fuese su predeterminado, y si la respuesta era afirmativa se generaba el cambio.

Además, en las opciones de usuario también tenía la posibilidad de cambiar de idioma predeterminado.

Cuando se realizaba un cambio de idioma, se renovaba la tabla de hash con las páginas cargadas, para disminuir la memoria utilizada.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Sin embargo, se deshabilitó la parte de euskara durante el desarrollo debido a que cada vez que había que realizar algún cambio en la aplicación había que implementarlo en ambas páginas, siendo muy costoso, además de necesitar a alguna persona experta que supiera realizar las traducciones oportunas.

Por ello nos dedicamos hasta el final del proyecto a implementar las nuevas funcionalidades y arreglo de bugs exclusivamente.

Ahora mismo está deshabilitado el idioma euskara tanto en la parte de hacer clic en los botones de la parte superior izquierda en la barra de color, como en las páginas principales de administración, datos o consulta donde se puede cambiar de idioma.

Sin embargo la aplicación está preparada para poder ser traducida en cualquier momento al euskara como a cualquier otro idioma si fuera necesario.

Lo único que se necesitaría hacer es como ya se ha comentado anteriormente, copiar las páginas exactas desde la carpeta /ES/ a la carpeta /EU/, traducir el contenido que el usuario visualiza y habilitar los botones de cambio de idioma.

4.6 INSERCIÓN DE FACTURAS DESDE XML

Como ya se ha comentado, ya existe un programa que analiza el fichero XML proporcionado por Endesa y lo introduce en el sistema de datos, así como otro que calcula los valores de la factura [1], pero como sucede con las conexiones SQL, el sistema de seguridad de Silverlight no nos permite una conexión directa con él, ni una manera sencilla de subir los archivos al servidor.

Para solucionarlo se ha desarrollado una página web en ASP.NET que incluye el programa. En esta página se proporciona al usuario una interfaz desde la que puede introducir el fichero así como los valores proporcionados en la factura.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

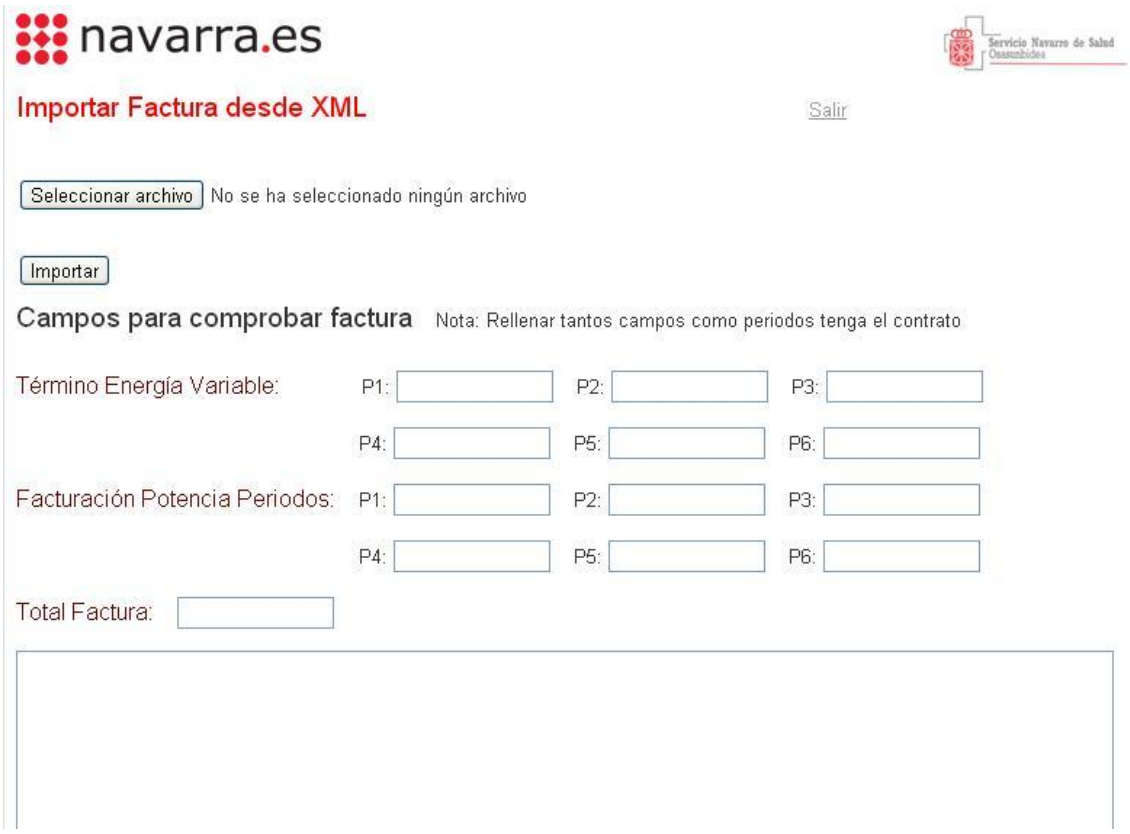


Figura 100: Inserción de Factura desde Archivo XML

Cuando el usuario pulsa el botón de inserción, el archivo es subido a un directorio temporal, utilizando un nombre aleatorio. En ese momento se aplica sobre él dicho programa de inserción y el de cálculo, obteniendo los datos de la factura.

Si los datos no coincidieran con los proporcionados por el usuario, los resultados se mostrarían indicando esta circunstancia, además de insertar una alarma en el sistema.

En cuanto a la página de comprobación de la Factura de Endesa, el funcionamiento es exacto a este caso, con la única diferencia de que en lugar de insertar un fichero XML, se debe indicar el número de factura a comprobar.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.6.1 SEGURIDAD

Solamente usuarios correctamente autentificados deberán poder acceder a la página de inserción o cálculo de facturas desde XML, por lo que debemos asegurarnos de que nadie más consigue entrar, pero volver a pedir el nombre de usuario y contraseña sería incómodo para los usuarios, por lo que ideamos un nuevo sistema.

Estas páginas únicamente serán accesibles desde la aplicación principal en Silverlight y no se podrá acceder a ellas de ninguna otra manera. Cuando un usuario selecciona el enlace de insertar factura desde XML, se lanza un proceso que generará en una tabla auxiliar de la Base de Datos una fila con:

- Identificador del usuario.
- Identificador del rol.
- Número aleatorio.

Así, cada vez que alguien quiera acceder se le asigna a su usuario y rol un número aleatorio y se almacena en la base de datos, y sólo una vez generado se abre una nueva ventana del navegador hacia la página de inserción.

```
/* Para insertar una factura desde xml vamos a otra página externa. Para garantizar la seguridad
 * asignamos a ese usuario un código único*/
[OperationContract]
public int insertXMLAccess(int userId, int roleId)
{
    DatabaseDataContext dc = new DatabaseDataContext();
    //Generamos un número aleatorio
    Random generator = new Random(DateTime.Now.Millisecond);
    int random;

    random = generator.Next(0, 100000000);

    //Insertamos en la tabla de seguridad
    string query = "INSERT INTO Seguridad VALUES({0},{1},{2});";

    dc.ExecuteCommand(query, userId, roleId, random);

    return random;
}
```

Figura 101: Inserción de Código de Seguridad

Para permitir el acceso, a la página de inserción se le pasan como variables GET los tres números que permiten el acceso. Al cargarse la página, ésta intenta eliminar en la base de datos la línea que los contiene. Si tiene éxito y puede borrarlo, garantiza el acceso al sistema y en caso contrario lo rechaza, redireccionando el navegador hacia una página de error.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//Obtener los valores
try
{
    usuario = int.Parse(Request.QueryString.Get("code"));
    rol = int.Parse(Request.QueryString.Get("access"));
    clave = int.Parse(Request.QueryString.Get("number"));
}
catch (Exception ex)
{
    return false;
}

con.Open();
con.CreateCommand();

string query = "DELETE FROM Seguridad WHERE idUsuario=@usuario AND idRol=@rol AND clave=@clave;";
SqlCommand cmd = new SqlCommand(query, con);

cmd.Parameters.Add(new SqlParameter("usuario", usuario));
cmd.Parameters.Add(new SqlParameter("rol", rol));
cmd.Parameters.Add(new SqlParameter("clave", clave));

Session["access"] = true;
Session.Timeout = 5;

deletedRows = cmd.ExecuteNonQuery();

con.Close();

if (deletedRows == 0)
    return false;
else
    return true;
```

Figura 102: Comprobación del Código de Seguridad

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.7 INSERCIÓN DE FACTURAS DESDE EXCEL

Hacia Marzo del 2012 se produjo por parte del Servicio Navarro de Salud – Osasunbidea un cambio de la compañía de suministro de energía contratada pasando de tener contratos con Endesa a Unión Fenosa.

Esta nueva compañía no proporciona ni la misma información acerca de las facturas ni en el mismo formato que Endesa. Ahora el nuevo formato donde nos presentan las facturas es en Excel y no nos dan ninguna información sobre los consumos por cuartos horarios, sino que únicamente nos indican los consumos por periodos. Además en el Excel existe información una o varias facturas, sin embargo en el XML sólo había información de una única factura.

Por ello es necesita tanto de un nuevo programa de inserción de facturas como una nueva interfaz que se adapte a las nuevas necesidades.

4.7.1 PROGRAMA

El objetivo de este programa es el chequeo de los archivos Excel que Unión Fenosa nos entrega con todas las facturas que el Servicio Navarra de Salud- Osasunbidea genera, para posteriormente cargar las facturas correctas en el sistema y no cargar las incorrectas.

Como es algo habitual que existan fallos en el Excel, y no se lleguen a cargar algunas facturas, ya existe en la aplicación una zona donde se pueden introducir manualmente facturas una por una.

MÉTODO CHEQUEAR

El método chequear se encarga de comprobar que la información del Excel sea coherente y no haya fallos de sumas, multiplicaciones, errores en los precios de las tarifas en comparación con los precios almacenados en la base de datos, error en el cálculo del IVA, del IEE, etc, realizando una conexión sobre él.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Esta función devuelve un entero según los eventos que se produzcan, un 0 si todas las facturas del Excel son correctas en cuanto a cálculos, 1 si hay facturas con errores en los cálculos, y -1 si ha ocurrido algún error inesperado.

```
//Devuelve 0 si todo los cálculos son correctos, 1 si hay facturas incorrectas de cálculos y -1 si hay error desconocido
public int chequear()
{
    System.Data.OleDb.OleDbConnection MyConnection;
    System.Data.DataSet ds;
    System.Data.OleDb.OleDbDataAdapter MyCommand;
    int filas, columnas, fila_etiquetas;
    int devuelve = 0;
    string nombre_hoja;
    DataTable tabla;

    //Siempre será la segunda fila, ya que la primera serán las cabeceras, aunque haya filas en blanco por encima de las cabeceras
    fila_etiquetas = 2;
    //Consulta consiguiendo todos los datos (cabeceras y datos)
    MyConnection = new System.Data.OleDb.OleDbConnection("provider=Microsoft.Jet.OLEDB.4.0;Data Source=" + archivo + ";Extended Properties=" + "\"Excel 8
    MyConnection.Open();
    tabla = new DataTable();
    tabla = MyConnection.GetOleDbSchemaTable(OleDbSchemaGuid.Tables,null);
```

Figura 103: Método Chequear

Este método está compuesto de llamadas a otros métodos que son realmente los que se encargan de realizar las comprobaciones y según sean correctos o no los cálculos irá guardando en los atributos toda la información necesaria de los errores producidos.

Ahora se van a analizar uno por uno cada una de las comprobaciones.

FALLO EN LOS CONTADORES

Realiza la comprobación que el consumo de cada periodo se corresponda con la resta entre el valor del contador final del periodo y el valor del contador inicial del periodo.

```
void CompruebaFalloContadores(OleDbDataReader dr, int j, int fila_etiquetas)
{
    double resta, valor1, valor2, total;
    int i = fila_etiquetas;
    SqlDataReader reader;
    SqlCommand command = con.CreateCommand();
    Double lecturaFinalBD,lecturaInicialExcel;
    DateTime fecha_ini_fact, fecha_fin_fact;
    string[] split_tarifa;
    string codTarifa, contrato;

    while (dr.Read())
    {
        //Inicializar valores
        codTarifa = "";
        contrato = "";
        fecha_ini_fact = DateTime.MinValue;
        fecha_fin_fact = DateTime.MinValue;

        //Si no son facturas estimadas
        if (!(Filas_fact_estimada.Contains(i)))
        {
            //Valores de inicio y fin
            valor1 = ConvDouble(dr["LEC_INI_ACT_P" + j].ToString());
            valor2 = ConvDouble(dr["LEC_FIN_ACT_P" + j].ToString());

            //Comprobamos que no haya fallo de lectura en los contadores
            resta = valor2 - valor1;

            //Porque hay algunos con decimales en los leidos y el consumido te lo ponen redondeado, considero que fallo a partir de en un decimal de marg
            if (((total = resta - ConvDouble(dr["CONSUMO_LE?DO ACT_P" + j].ToString())) >= 1.0 || (total = resta - ConvDouble(dr["CONSUMO_LE?DO ACT_P" +
            {
                //Si tiene ya la fila de la factura mal
                if (!Filas_chequeo_fallo.Contains(i))
                {
                    //Insertamos la fila del error
                    Filas_chequeo_fallo.Add(i);
                    //Insertamos el número de factura correspondiente a la fila
                    Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
                    //Insertamos su correspondiente error
                    String_chequeo_fallo.Add("en los contadores de consumo, en el periódo " + j);
```

Figura 104: Método Fallo Contadores

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Además tiene una parte de código extra (para aprovechar que estamos comprobando los contadores) que chequea si hay un solapamiento entre el valor del contador de cada periodo de la última factura insertada en la base de datos y la lectura inicial del contador de cada periodo de la factura del Excel, es decir, consigue el valor de la lectura final del contador de la última factura de cada periodo insertada en la base de datos y comprueba si el valor de la lectura del contador inicial del periodo continua en el mismo número, hay un hueco o se solapa.

```
//Seleccionamos la lectura final de la última factura insertada de ese contrato y tarifa, en esas fechas y periodo
command.CommandText = "SELECT lecFin " +
                      "FROM Factura " +
                      "JOIN ConsAct on ConsAct.idFactura = Factura.idFactura " +
                      "JOIN Contrato on Contrato.idContrato=Factura.Contrato_idcontrato " +
                      "JOIN Tarifa on Tarifa.idTarifa = Contrato.Tarifa_idTarifa " +
                      "JOIN TipoTarifa on TipoTarifa.idTipoTarifa = tarifa.TipoTarifa_idTipoTarifa " +
                      "JOIN Asoc_Periodos_Tipo on Asoc_Periodos_Tipo.idTipoTarifa = TipoTarifa.idTipoTarifa " +
                      "JOIN PeriodosTarifa on PeriodosTarifa.idPeriodosTarifa = Asoc_Periodos_Tipo.idPeriodosTarifa " +
                      "WHERE CONVERT(int,SUBSTRING(PeriodosTarifa.Periodo,Charindex(' ', PeriodosTarifa.Periodo)+1,LEN(PeriodosTarifa.Periodo))) = ConsAct.pe
                      "AND FechaIni_Tarifa <= CONVERT(datetime,'" + fecha_ini_fact + "',103) " +
                      "AND (FechaFin_Tarifa IS NULL OR(CONVERT(datetime,'" + fecha_ini_fact + "',103) <= FechaFin_Tarifa AND CONVERT(datetime,'" + fecha_fin_
                      "AND FechaIni_Contrato <= CONVERT(datetime,'" + fecha_ini_fact + "',103) " +
                      "AND (FechaFin_Contrato IS NULL OR (CONVERT(datetime,'" + fecha_ini_fact + "',103)<= FechaFin_Contrato AND CONVERT(datetime,'" + fecha_
                      "AND TARIFA.idTarifa = Asoc_Periodos_Tipo.Asoc_Periodos_Tipo_idTarifa " +
                      "AND numcontrato= '" + contrato + "' " +
                      "AND CodTarifa= '" + codTarifa + "' " +
                      "AND ConsAct.periodo = "+ j +" " +
                      "AND FechaFin_Factura<= CONVERT(datetime,'" + fecha_ini_fact + "',103) ";

reader = command.ExecuteReader();
//Si existe alguna factura comprobamos, si no nada
if (reader.HasRows)
{
    reader.Read();

    lecturaFinalBD = Convert.ToDouble(reader["lecFin"].ToString());
    lecturaInicialExcel = Convert.ToDouble(dr["LEC_INI_ACT_P" + j].ToString());

    //Si se solapan
    if ((lecturaInicialExcel - lecturaFinalBD) < 0)
    {
        //Si tiene ya la fila de la factura mal
        if (!Filas_chequeo_fallo.Contains(i))
        {
            //Insertamos la fila del error
            Filas_chequeo_fallo.Add(i);
            //Insertamos el número de factura correspondiente a la fila
            Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
            //Insertamos su correspondiente error
            String_chequeo_fallo.Add("hay solapamiento en las lecturas de los contadores de consumo (última lectura en la BD: " + lecturaFinalBD + ", lectura
        }
    }
```

Figura 105: Chequeo solapamientos facturas

FALLO EN LA SUMA DE EQUIVALENCIAS DE LOS PERIODOS

Comprueba la equivalencia entre los consumos leídos y el paso a consumos contratados, ya que siempre se leen los consumos de los 6 periodos pero realmente se tienen contratos de tarifas de 1, 2, 3 o 6 periodos, luego la suma total de ambos cálculos debe ser la misma.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
//Comprueba la equivalencia entre los consumos leídos (6 periodos) y el paso a consumos contratados (2,3 o 6)
void CompruebaFalloSuma(OleDbDataReader dr, int fila_etiquetas)
{
    double total, suma1, suma2;
    int i = fila_etiquetas;
    while (dr.Read())
    {
        suma1 = ConvDouble(dr["CONSUMO_LE?DO ACT_P1"].ToString()) + ConvDouble(dr["CONSUMO_LE?DO ACT_P2"].ToString()) + ConvDouble(dr["CONSUMO_LE?DO ACT_P3"]
        suma2 = ConvDouble(dr["CONS ACT FACT P1"].ToString()) + ConvDouble(dr["CONS ACT FACT P2"].ToString()) + ConvDouble(dr["CONS ACT FACT P3"].ToString())
        //Porque los consumidos son redondeados, y al pasar a 3 periodos considero que fallo a partir de 2 de margen y si lo que te cobran es menor que lo qu
        if (((total = suma1 - suma2) >= 2.0 || (total = suma1 - suma2) <= -2.0) && (suma2 > suma1))
        {
            //Si tiene ya la fila de la factura mal
            if (!Filas_chequeo_fallo.Contains(i))
            {
                //Insertamos la fila del error
                Filas_chequeo_fallo.Add(i);
                //Insertamos el número de factura correspondiente a la fila
                Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
                //Insertamos su correspondiente error
                String_chequeo_fallo.Add("en la equivalencia entre la suma de los consumos leídos (6 periodos: " + suma1 + ") y la suma del paso a consumos c
            }
        }
        i++;
    }
    //Cerramos el lector
    dr.Close();
}
```

Figura 106: Método Comprueba Fallo en las Equivalencias de los Periodos

FALLO EN LA MULTIPLICACIÓN DE CONSUMOS POR PRECIO

Chequea que sea correcto el cálculo del importe de cada periodo total, siendo resultado de la multiplicación entre el consumo realizado y el precio correspondiente de cada periodo.

```
//Comprueba la multiplicacion de consumos por precio
void CompruebaFalloMultiplicacion(OleDbDataReader dr, int fila_etiquetas)
{
    double total1, total2, total3, total4, total5, total6;
    int i = fila_etiquetas;
    while (dr.Read())
    {
        //ImporteTotal - Consumo * precio
        total1 = (ConvDouble(dr["IMP ACT P1"].ToString())) - (ConvDouble(dr["CONS ACT FACT P1"].ToString()) * ConvDouble(dr["ACT_PRECIO_P1"].ToString()));
        total2 = (ConvDouble(dr["IMP ACT P2"].ToString())) - (ConvDouble(dr["CONS ACT FACT P2"].ToString()) * ConvDouble(dr["ACT_PRECIO_P2"].ToString()));
        total3 = (ConvDouble(dr["IMP ACT P3"].ToString())) - (ConvDouble(dr["CONS ACT FACT P3"].ToString()) * ConvDouble(dr["ACT_PRECIO_P3"].ToString()));
        total4 = (ConvDouble(dr["IMP ACT P4"].ToString())) - (ConvDouble(dr["CONS ACT FACT P4"].ToString()) * ConvDouble(dr["ACT_PRECIO_P4"].ToString()));
        total5 = (ConvDouble(dr["IMP ACT P5"].ToString())) - (ConvDouble(dr["CONS ACT FACT P5"].ToString()) * ConvDouble(dr["ACT_PRECIO_P5"].ToString()));
        total6 = (ConvDouble(dr["IMP ACT P6"].ToString())) - (ConvDouble(dr["CONS ACT FACT P6"].ToString()) * ConvDouble(dr["ACT_PRECIO_P6"].ToString()));
        //Si se pasa en 0.1 lo considero fallo
        if ((total1 >= 0.1 || total1 <= -0.1) || (total2 >= 0.1 || total2 <= -0.1) || (total3 >= 0.1 || total3 <= -0.1) || (total4 >= 0.1 || total4 <= -0.1)
        {
            //Si tiene ya la fila de la factura mal
            if (!Filas_chequeo_fallo.Contains(i))
            {
                //Insertamos la fila del error
                Filas_chequeo_fallo.Add(i);
                //Insertamos el número de factura correspondiente a la fila
                Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
                //Insertamos su correspondiente error
                String_chequeo_fallo.Add("en la multiplicación de consumos por precio");
            }
        }
        i++;
    }
    //Cerramos el lector
    dr.Close();
}
```

Figura 107: Método Comprueba Fallo en las Multiplicaciones

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

FALLO EN LA ACTIVA TOTAL

Comprueba que la suma de todos los consumos de todos los periodos junto con el incremento de PEA se corresponda con el importe total de la activa.

```
//Comprueba el calculo del importe de la activa total
void CompruebaActivaTotal(OleDbDataReader dr, int fila_etiquetas)
{
    double total;

    int i = fila_etiquetas;
    while (dr.Read())
    {
        //Resta Activa total menos Importes Activas + Importe Incremento Energia PEA
        total = ConvDouble(dr["IMP_ACT_TOTAL"].ToString()) - (ConvDouble(dr["IMP ACT P1"].ToString()) + ConvDouble(dr["IMP ACT P2"].ToString()) + ConvDouble(
        //Si se pasa en 0.1 lo considero fallo
        if ((total >= 0.1 || total <= -0.1))
        {
            //Si tiene ya la fila de la factura mal
            if (!Filas_chequeo_fallo.Contains(i))
            {
                //Insertamos la fila del error
                Filas_chequeo_fallo.Add(i);
                //Insertamos el número de factura correspondiente a la fila
                Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
                //Insertamos su correspondiente error
                String_chequeo_fallo.Add("en el cálculo del importe de la activa total");
            }
        }
        i++;
    }
    //Cerramos el lector
    dr.Close();
}
```

Figura 108: Método Comprueba Activa Total

FALLO EN EL CÁLCULO DEL IVA Y DEL IEE

Realiza la comprobación del correcto cálculo del IVA y del Impuesto Eléctrico (IEE).

```
//REALIZAMOS TODOS LOS CÁLCULOS
totalIEE = (ConvDouble(dr["IMP_ACT_TOTAL"].ToString()) + ConvDouble(dr["IMP_REACT"].ToString()) + ConvDouble(dr["IMP_EXCESO_POT"].ToString()) + ConvDouble
total1 = totalIEE - (ConvDouble(dr["IMP IEE"].ToString()));
totalIVA = (ConvDouble(dr["IMP_ACT_TOTAL"].ToString()) + ConvDouble(dr["IMP_REACT"].ToString()) + ConvDouble(dr["IMP_EXCESO_POT"].ToString()) + ConvDouble
total2 = totalIVA - (ConvDouble(dr["IMP_IVA_IGIC"].ToString()));
//Cálculo IEE. Porque hay algunos con decimales que los ponen redondeados, considero que fallo a partir de en un decimal de margen
if (total1 > 0.01 || total1 < -0.01)
{
    //Si tiene ya la fila de la factura mal
    if (!Filas_chequeo_fallo.Contains(i))
    {
        //Insertamos la fila del error
        Filas_chequeo_fallo.Add(i);
        //Insertamos el número de factura correspondiente a la fila
        Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
        //Insertamos su correspondiente error
        String_chequeo_fallo.Add("en el cálculo del IEE, en el excel es " + (ConvDouble(dr["IMP IEE"].ToString())) + " y en el cálculo es " + totalIEE);
    }
}
//Cálculo IVA. Porque hay algunos con decimales que los ponen redondeados, considero que fallo a partir de en un decimal de margen
if (total2 > 0.01 || total2 < -0.01)
{
    //Si tiene ya la fila de la factura mal
    if (!Filas_chequeo_fallo.Contains(i))
    {
        //Insertamos la fila del error
        Filas_chequeo_fallo.Add(i);
        //Insertamos el número de factura correspondiente a la fila
        Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
        //Insertamos su correspondiente error
        String_chequeo_fallo.Add("en el cálculo del IVA, en el excel es " + (ConvDouble(dr["IMP_IVA_IGIC"].ToString())) + " y en el cálculo es " + totalIV
    }
}
```

Figura 109: Método Comprueba IVA e IEE

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

FALLO SUMA TOTAL DE LA FACTURA

Se encarga de chequear que la suma de los importes e impuestos sean equivalentes al valor total de la factura.

```
//REALIZAMOS TODOS LOS CÁLCULOS
totalIEE = (ConvDouble(dr["IMP_ACT_TOTAL"].ToString()) + ConvDouble(dr["IMP_REACT"].ToString()) + ConvDouble(dr["IMP_EXCESO_POT"].ToString()) + ConvDouble
total1 = totalIEE - (ConvDouble(dr["IMP IEE"].ToString()));
totalIVA = (ConvDouble(dr["IMP_ACT_TOTAL"].ToString()) + ConvDouble(dr["IMP_REACT"].ToString()) + ConvDouble(dr["IMP_EXCESO_POT"].ToString()) + ConvDouble
total2 = totalIVA - (ConvDouble(dr["IMP_IVA_IGIC"].ToString()));
//Cálculo IEE. Porque hay algunos con decimales que los ponen redondeados, considero que fallo a partir de en un decimal de margen
if (total1 > 0.01 || total1 < -0.01)
{
    //Si tiene ya la fila de la factura mal
    if (!Filas_chequeo_fallo.Contains(i))
    {
        //Insertamos la fila del error
        Filas_chequeo_fallo.Add(i);
        //Insertamos el número de factura correspondiente a la fila
        Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
        //Insertamos su correspondiente error
        String_chequeo_fallo.Add("en el cálculo del IEE, en el excel es " + (ConvDouble(dr["IMP IEE"].ToString())) + " y en el cálculo es " + totalIEE);
    }
}
//Cálculo IVA. Porque hay algunos con decimales que los ponen redondeados, considero que fallo a partir de en un decimal de margen
if (total2 > 0.01 || total2 < -0.01)
{
    //Si tiene ya la fila de la factura mal
    if (!Filas_chequeo_fallo.Contains(i))
    {
        //Insertamos la fila del error
        Filas_chequeo_fallo.Add(i);
        //Insertamos el número de factura correspondiente a la fila
        Numero_facturas_chequeo_fallo.Add(dr["NUMERO_FACTURA"].ToString());
        //Insertamos su correspondiente error
        String_chequeo_fallo.Add("en el cálculo del IVA, en el excel es " + (ConvDouble(dr["IMP_IVA_IGIC"].ToString())) + " y en el cálculo es " + totalI\
    }
}
```

Figura 110: Método Comprueba Suma Total

FALLO POTENCIA

Este método realiza la comprobación de que la potencia que Excel indica que tiene el contrato de la factura a chequear se corresponde con la potencia almacenada en la base de datos.

```
//Comprueba que la potencia contratada del contrato de la BD sea igual a la que indica el excel
void CompruebaPotenciaContratada(OleDbDataReader dr, int fila_etiquetas)
{
    SqlDataReader reader;
    SqlCommand command = con.CreateCommand();
    string contrato, CUPS, codTarifa;
    string[] split_tarifa;
    DateTime fecha_ini_fact, fecha_fin_fact;
    double potenciaBD, total;
    double potEx;
    int filas = 0;
    int i = fila_etiquetas;
    while (dr.Read())
    {
```

Figura 111: Método Comprueba Potencia

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

FALLO PRECIOS TARIFA

Comprueba que los precios de la tarifa del Excel se corresponden con los precios almacenados en la base de datos.

```
//Comprueba que los precios de la Energía activa del contrato de la BD sea igual a los precios del excel
void CompruebaPreciosActiva(OleDbDataReader dr, int fila_etiquetas)
{
    SqlDataReader reader;
    SqlCommand command = con.CreateCommand();
    double precioEx, precioBD, total;
    int idContrato, Tarifa_idTarifa, TUR_idTUR, periodos;
    int filas, num_contratos = 0;
    int i = fila_etiquetas;
    string contrato, CUPS,codTarifa;
    string[] split_tarifa;
    DateTime fecha_ini_fact, fecha_fin_fact;

    //Recorriendo cada fila del excel
    while (dr.Read())
    {
```

Figura 112: Método Comprueba Precios

FACTURA ESTIMADA O SUMINISTRO DE SOCORRO

Una factura estimada consiste tal y como indica su nombre en una factura cuyo consumo no ha sido contabilizado realmente en el contador sino que se ha hecho una estimación de cuánto debería de gastar en ese periodo de la factura según los consumos de las facturas anteriores.

Un suministro de socorro se trata de aquellos cuyo consumo son 0 salvo acciones puntuales, se usan únicamente cuando algún suministro deja de funcionar por cualquier motivo o se necesita de otro, es decir, como auxiliar.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Lo que este método chequea es de si la factura comprobada se trata de una factura estimada o de si se trata de un suministro de socorro.

```
for (int x = 1; x <= 6; x++)
{
    //Primero de todo comprobar si son suministros de socorro ( LECTURA INICIAL >=0, LECTURA FINAL>=0 Y CONSUMO==0 )
    if ((ConvDouble(dr["LEC_INI_ACT_P" + x].ToString())) >= 0 && (ConvDouble(dr["LEC_FIN_ACT_P" + x].ToString())) >= 0 && (ConvDouble(dr["CONSUMO
    {
        //Si tiene ya la fila de la factura mal y en todos los periodos se cumplen las condiciones
        if (!Filas_sum_socorro.Contains(i) && contador == 6)
        {
            //Insertamos la fila del error
            Filas_sum_socorro.Add(i);
            //Insertamos el número de factura correspondiente a la fila
            Cups_sum_socorro.Add(dr["PRIMARY_CUPS"].ToString());

            //Actualización en la BD (sólo en caso que ese CUPS no sea un suministro de socorro)
            CUPS = dr["PRIMARY_CUPS"].ToString();
            if (CUPS_en_BD(CUPS) == 1)
            {
                CUPS = CUPS.Substring(0, 20);
            }
            CUPS = CUPS_espacios(CUPS); //Le añadimos los espacios en blanco
            command.CommandText = "IF(EXISTS (SELECT * FROM CUPS WHERE CUPS='" + CUPS + "' AND SumSocorro=0)) " +
                                  "UPDATE CUPS SET SumSocorro=1 WHERE CUPS='" + CUPS + "';";
            command.ExecuteNonQuery();
        }
        //Para sólo insertar suministro de socorro si todas las lecturas iniciales y finales son > 0  y los consumos = 0
        contador++;
    }
    //Segundo de todo comprobar si son facturas estimadas ( LECTURA INICIAL=0, LECTURA FINAL=0 Y CONSUMO>0 )
    else if ((ConvDouble(dr["LEC_INI_ACT_P" + x].ToString())) == 0 && (ConvDouble(dr["LEC_FIN_ACT_P" + x].ToString())) == 0 && (ConvDouble(dr["CO
    {
        //Si tiene al menos un consumo mayor que 0 y lectura a 0 de ese periodo ya la consideramos estimada
        if (!Filas_fact_estimada.Contains(i))
        {
            //Insertamos la fila del error
            Filas_fact_estimada.Add(i);
            //Insertamos el número de factura correspondiente a la fila
            Numero_fact_estimada.Add(dr["NUMERO_FACTURA"].ToString());
        }
    }
```

Figura 113: Método Chequea Suministro Socorro y Factura Estimada

MÉTODO CARGAR

Este método tiene la función de cargar en la base de datos lo necesario de todas aquellas facturas que no tengan ningún tipo de error o fallo en el chequeo anteriormente realizado.

Como posteriormente se explicará, las facturas se pueden almacenar en unas tablas u otras en la base de datos, ya que existe una funcionalidad que permite cargar las facturas en las tablas reales de la base de datos o guardar las facturas en tablas auxiliares almacenándolas temporalmente como borrador.

Esta función devuelve un entero según los eventos que se produzcan, un 0 si todas las facturas han sido insertadas en la base de datos sin problemas, 1 si hay facturas que no han conseguido ser insertadas, y -1 si ha ocurrido algún error inesperado.

Al igual que el método de chequear irá guardando en los atributos toda la información referente a los errores que se produzcan.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
consumoAct = double.Parse(dr["CONS ACT FACT P" + i].ToString());
maximoAct = double.Parse(dr["POT_MAX_DEMAN_ATR_" + i].ToString());
lecIni = double.Parse(dr["LEC_INI_ACT_P" + i].ToString());
lecFin = double.Parse(dr["LEC_FIN_ACT_P" + i].ToString());

command.Parameters.AddWithValue("@param1", consumoAct);
command.Parameters.AddWithValue("@param2", maximoAct);
command.Parameters.AddWithValue("@param4", lecIni);
command.Parameters.AddWithValue("@param5", lecFin);
//Real
if (importa_o_borrador == 0)
{
    command.CommandText = "INSERT INTO ConsAct(idFactura,periodo,consumoAct,maximoAct,lecIni,lecFin,aciAct) VALUES (" + idFactura + "," + i + ",@param1,@
    command.ExecuteNonQuery();
    command.Parameters.AddWithValue("@param3", consumoReact);
    command.CommandText = "INSERT INTO ConsReact(idFactura,periodo,consumoReact) VALUES (" + idFactura + "," + i + ",@param3)";
    command.ExecuteNonQuery();
}//Borrador
else if (importa_o_borrador == 1)
{
    command.CommandText = "INSERT INTO ConsAct_Aux(idFactura,periodo,consumoAct,maximoAct,lecIni,lecFin,aciAct) VALUES (" + idFactura + "," + i + ",@para
    command.ExecuteNonQuery();
    command.Parameters.AddWithValue("@param3", consumoReact);
    command.CommandText = "INSERT INTO ConsReact_Aux(idFactura,periodo,consumoReact) VALUES (" + idFactura + "," + i + ",@param3)";
    command.ExecuteNonQuery();
}
```

Figura 114: Método Cargar

ATRIBUTOS DE ALMACENAMIENTO DE ERRORES

Para poder guardar toda la información referente a los errores que se produzcan tanto en el chequeo como en la carga de facturas, se han creado una serie de atributos.

Atributos relacionados con el chequeo:

- filas_chequeo_fallo: línea del fallo en el Excel
- numero_facturas_chequeo_fallo: número de la factura en la que hay error
- string_chequeo_fallo: información del fallo producido

Atributos relacionados con la carga:

- filas_facturas_fallo: línea del fallo en el Excel
- numero_facturas_fallo: número de la factura en la que hay error
- string_facturas_fallo: información del fallo producido

También se necesita saber si una factura es estimada o de si el CUPS a la que pertenece una factura es un suministro de socorro, para luego mostrar información al usuario.

Atributos relacionados con las facturas estimadas:

- filas_fact_estimada: línea del Excel
- numero_fact_estimada: número de la factura estimada

Atributos relacionados con los suministros de socorro

- filas_sum_socorro: línea del Excel
- cups_sum_socorro: CUPS al que pertenece la factura

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.7.2 INTERFAZ

Al igual que en la inserción de facturas desde un XML se ha tenido que crear un interfaz que incorpore el programa de inserción de facturas a partir de un Excel.

Se ha desarrollado de la misma manera, una página web en ASP.NET. En esta página se le proporciona al usuario una interfaz desde la que puede introducir el fichero con facturas a cargar en la base de datos.

Figura 115: Inserción de Facturas desde Archivo Excel

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Primeramente el usuario tiene que seleccionar qué archivo quiere cargar.

Segundo, existen dos maneras de cargar las facturas:

- El usuario pulsa el botón de importar directamente, el archivo es subido a un directorio temporal, utilizando un nombre aleatorio. En ese momento se aplica sobre él dicho programa de chequeo y carga, insertándose en la base de datos las facturas correctas. Finalmente se generará en C:\Temp otro fichero Excel similar al archivo subido, sustituyendo los datos de tres columnas innecesarias por: una columna en la que se muestra el error del cálculo en Excel rellanada en color rojo, y si es correcto en verde, otra columna igual pero mostrando el error en la carga de la factura en la base de datos, sino en verde también y en la tercera columna informa en color azul o en naranja al usuario si esa factura se trata de un suministro de socorro o es una factura estimada correspondientemente.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	MES_EMISI?	UMERO_FACT	de cálculo e	ar facturas es	Socorro o Es	PRIMARY_CU	Contrato	Clave Agrupa	DIRECCION_SUMINISTRO	DISTRIBUIDO	SECTOR	TARIFA_ACCE	ECHA_EN
2	01/08/2012	0312025000373	en el cálculo	esta factura		ES0021000006	031202001068	GG003	AVDA TARAZONA , 121 PK3 - NAVARRA TUDELA (31500)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	02/08
3	01/08/2012	0312025000375	Correcto	es un número	Sum.Socorro	ES0021000006	031202001068	GG003	AVDA TARAZONA , 121 PK3 - NAVARRA TUDELA (31500)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	08/08
4	01/08/2012	0312025000375	no existe el	esta factura		ES0021000006	031202001057	GG003	IRUNLARREA,4 - Bj -NAVARRA PAMPLONA/IRU?A(31008)	DISTRIBUCI?N	Electricidad	Acceso 6.1 (1	08/08
5	01/08/2012	0312025000375	no existe el	esta factura		ES0021000006	031202001072	GG003	CALLE SANTA SORIA , 22 BA2 - NAVARRA ESTELLA/LIZARRA	DISTRIBUCI?N	Electricidad	Acceso 6.1 (1	08/08
6	01/08/2012	0312025000376	no existe el	esta factura		ES0021000006	031202001075	GG003	CTRA AOIZ,S/N -NAVARRA EG?ES(31486)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	08/08
7	01/08/2012	0312025000384	no existe el	esta factura		ES0021000006	031202001064	GG003	AVDA VILLAVA / ATARRABIAKO ETORBIDEA , 53 - NAVARRA	DISTRIBUCI?N	Electricidad	Acceso 3.1A	31/08
8	01/08/2012	0312034000201	no existe el	esta factura		ES0021000006	031202001072	GG003	CALLE SANTA SORIA , 22 BA2 - NAVARRA ESTELLA/LIZARRA	DISTRIBUCI?N	Electricidad	Acceso 6.1 (1	08/08
9	01/08/2012	0312034000201	no existe el	esta factura		ES0021000006	031202001057	GG003	IRUNLARREA,4 - Bj -NAVARRA PAMPLONA/IRU?A(31008)	DISTRIBUCI?N	Electricidad	Acceso 6.1 (1	08/08
10	01/08/2012	0312034000201	no existe el	esta factura		ES0021000006	031202001071	GG003	AVDA TARAZONA , 121 KM3 - NAVARRA TUDELA (31500)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	08/08
11	01/08/2012	0312034000201	la potencia	esta factura		ES0021000006	031202001068	GG003	AVDA TARAZONA , 121 PK3 - NAVARRA TUDELA (31500)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	08/08
12	01/08/2012	0312034000201	no existe el	esta factura		ES0021000006	031202001075	GG003	CTRA AOIZ,S/N -NAVARRA EG?ES(31486)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	08/08
13	01/08/2012	0312035000333	no existe el	esta factura		ES0021000006	031202001071	GG003	AVDA TARAZONA , 121 KM3 - NAVARRA TUDELA (31500)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	08/08
14	01/08/2012	0312041040908	no existe el	esta factura		ES0021000006	031202001057	GG003	IRUNLARREA,4 - Bj -NAVARRA PAMPLONA/IRU?A(31008)	DISTRIBUCI?N	Electricidad	Acceso 6.1 (1	09/08
15	01/08/2012	0312041040908	Correcto	es un número		ES0021000006	031202001068	GG003	AVDA TARAZONA , 121 PK3 - NAVARRA TUDELA (31500)	DISTRIBUCI?N	Electricidad	Acceso 6.2 (36	09/08

Figura 116: Excel generado con fallos de cálculo

- El usuario pulsa el botón de guardar en borrador, el archivo es subido a un directorio temporal. Se aplica el chequeo y se genera el Excel en un directorio temporal, posteriormente se aplica la carga de facturas pero en vez de cargarse las facturas en las tablas reales de la base de datos, se cargan en tablas auxiliares. El interfaz cambiará de estado si todo ha ido correcto y ahora el usuario sólo podrá pinchar en el botón de eliminar el borrador (elimina las facturas cargadas anteriormente) o pinchar en el botón de validar, lo cual hará que se muevan las facturas de las tablas auxiliares de la base de datos a las tablas reales, al igual que el Excel, se moverá desde el directorio temporal a C:\Temp.

Figura 117: Opción de guardar en borrador

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

De cualquiera de las maneras, en los cuadros de texto se mostrará información al usuario acerca de todos los errores que hayan surgido en rojo o si ha funcionado todo bien, aparecerán comentarios en verde.

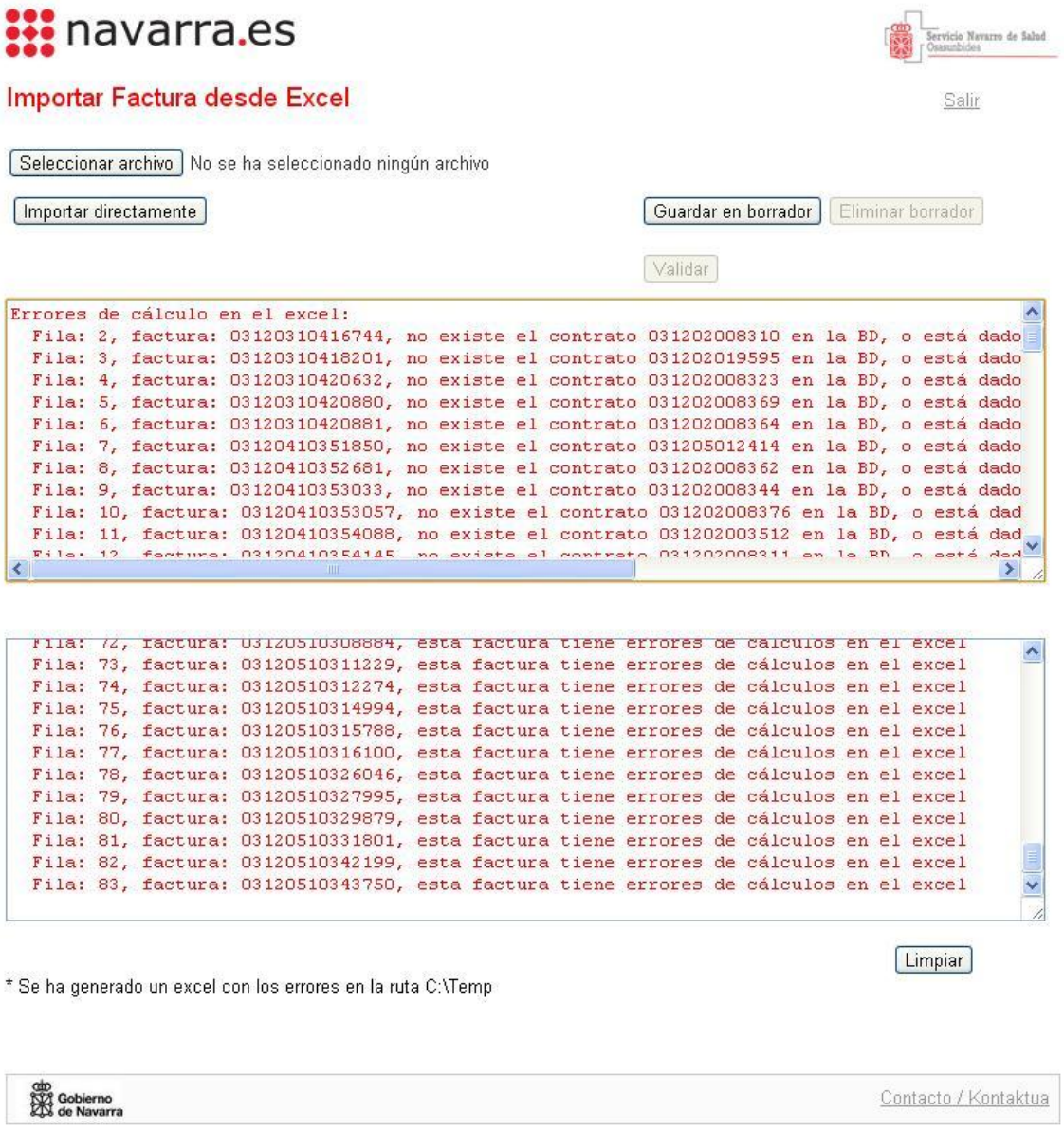


Figura 118: Errores en los cuadros de texto

Finalmente, el comportamiento de cargar las facturas directamente o cargarlas guardándolas en borrador es parecido en cuanto a código, sin embargo a afectos de visualización del usuario es totalmente diferente. Se decidió implementar la carga en borrador para que el usuario tenga la posibilidad de que si en un determinado momento no puede analizar las facturas cargadas (le llega la hora de comer o es la hora de irse a casa) y necesita cargarlas, las almacene en borrador y en el momento que disponga de tiempo sepa que tiene que validar las que anteriormente almacenó, así no se le olvida.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.7.3 SEGURIDAD

La seguridad en la inserción de facturas de Unión Fenosa a partir de un fichero Excel, funciona exactamente igual que la ya explicada anteriormente en inserción de facturas de Endesa a partir de un archivo XML.

Solamente aquellos usuarios correctamente autentificados podrán acceder a esta página.

4.8 CONVERSIONES Y FUNCIONES AUXILIARES

Para mayor comodidad en el trabajo, se han programado una serie de funciones auxiliares que evitan la duplicidad de código y facilitan la programación. Se explica a continuación cada una de ellas.

4.8.1 DOBLE CLIC

Actualmente Silverlight no implementa ningún evento de doble clic sobre un elemento, por lo que se debe crear un sistema propio para comprobarlo. Se ha generado una clase llamada DoubleClick, que comprueba mediante la llamada a un método isDoubleClick() si ha sido un evento de doble clic o no.

```
public class DoubleClick
{
    private DateTime lastClick;
    private Boolean firstClickDone;
    private Point clickPosition;

    public DoubleClick()
    {
        //Variables
        lastClick = new DateTime();
        firstClickDone = false;
        clickPosition = new Point();
    }

    //Comprobar si es un doble click
    public bool isDoubleClick(UIElement sender, MouseButtonEventArgs e)
    {
        DateTime clickTime = DateTime.Now; //Ahora
        TimeSpan span = clickTime - lastClick; //Diferencia entre el primer click y ahora

        //Si la diferencia es mayor que 300 ms o es el primer click
        if (span.TotalMilliseconds > 300 || firstClickDone == false)
        {
            firstClickDone = true; //Activamos el primer click

            lastClick = DateTime.Now; //Último click = ahora
            return false; //No es doble click
        }

        else //En otro caso puede ser doble click
        {
            Point position = e.GetPosition(sender);

            //Si no hemos movido demasiado el ratón (4 puntos en cualquier eje)
            if (Math.Abs(clickPosition.X - position.X) < 4 && Math.Abs(clickPosition.Y - position.Y) < 4)
            {
                firstClickDone = false;
                return true; //Es un doble click
            }

            else //Si lo hemos movido no es un doble click
                return false;
        }
    }
```

Figura 119: Clase DoubleClick

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.8.2 FUNCIONES PARA LA COPIA DE OBJETOS

Los objetos de Silverlight no implementan por defecto ningún método que permita obtener una copia de ellos, a pesar de ser un lenguaje de programación orientado a objetos y basado en C#. Además de esto, los objetos obtenidos de la base de datos no pueden tener ningún método propio ni constructor para su correcto funcionamiento, por lo que se ha creado una clase estática para hacer la copia de aquellos objetos que se ha necesitado

La clase implementa el método copyObject, sobrecargado con cada tipo de objeto que queramos copiar como parámetro de entrada. Como parámetro de salida se obtiene la copia.

```
//Función para copiar objetos
public static ListLevel4 copyObject(ListLevel4 source)
{
    ListLevel4 returnValue = new ListLevel4();
    returnValue.idNivel4 = source.idNivel4;
    returnValue.Nombre = source.Nombre;
    returnValue.PorcentajeOcup = source.PorcentajeOcup;
    returnValue.idNivel3 = source.idNivel3;

    return returnValue;
}
public static ListLevel3 copyObject(ListLevel3 source)
{
    ListLevel3 returnValue = new ListLevel3();
    returnValue.idNivel3 = source.idNivel3;
    returnValue.Nombre = source.Nombre;
    returnValue.Direccion = source.Direccion;
    returnValue.CP = source.CP;
    returnValue.Localidad = source.Localidad;
    returnValue.Tipo = source.Tipo;
    returnValue.idNivel2 = source.idNivel2;

    return returnValue;
}
```

Figura 120: Método copyObject()

4.8.3 SELECTOR DE NAVEGACIÓN DE CUPS

Para el funcionamiento del selector de Organizaciones y CUPS, que nos permitía navegar por los distintos niveles y efectuar las asociaciones entre distintas tablas, se ha generado la clase OrgSelector.

Esta clase contiene los métodos:

selectorChange(): cuando la lista desplegable sobre la lista de organizaciones cambia, nos muestra las organizaciones para ese nivel.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

oneLevelDown(): avanza un nivel en la jerarquía de organizaciones.
oneLevelUp(): retrocede un nivel en la jerarquía de organizaciones.

4.8.4 CUPS

Un código CUPS debe mantener un formato específico, a saber, el identificador ES, seguido de un espacio, seguido de 4 números seguido de 3 grupos de 4 números separados entre sí por espacios, un grupo de 2 letras en mayúsculas (de control, calculados automáticamente a partir de los 16 numéricos anteriores), y como caracteres opcionales 1 carácter numérico, seguido de espacio y una letra mayúscula. Dos casos de ejemplos de CUPS:

- ES 1234 1234 5678 9012 JY 1 F
- ES 0987 5432 1098 7654 ZF

Para comprobar si el código mantiene este formato, se ha utilizado el siguiente método: que se apoya en el uso de expresiones regulares:

```
/* Comprueba si un string representa un CUPS. El formato del cups es: ES, espacio, 1 grupo de 4 números (identifica la distribuidora),
 * espacio, 3 grupos de 4 números separados por espacios (de libre asignación por el distribuidor), espacio,
 * 2 carácteres (alfabéticos en Mayúsculas) de control (calculados a partir de los 16 numéricos anteriores), espacio,
 * 1 carácter numérico(opcional), espacio, 1 carácter alfabético en mayúscula(opcional). Por ejemplo:
 * ES 1234 1234 5678 9012 JY 1 F
 * ES 0987 5432 1098 7654 ZF
 * Usaremos expresiones regulares para comprobarlo. */
public static bool isCUPS(string CUPS)
{
    //O tiene los opcionales o no los tiene
    Regex expression = new Regex("^ES [0-9]{4} [0-9]{4} [0-9]{4} [0-9]{4} [A-Z]{2} [0-9]{1} [A-Z]{1}$|^ES [0-9]{4} [0-9]{4} [0-9]{4} [0-9]{4} [A-Z]{2}$");

    return expression.IsMatch(CUPS);
}
```

Figura 121: Comprobación de Formato de CUPS

4.8.5 OTRAS

Además de éstos, se han desarrollado gran cantidad de métodos para comprobación de formatos (comprobar si es un valor numérico, comprobar longitudes de datos) o conversiones de cadenas de texto a valores numéricos, utilizados sobre todo a la hora de revisar si los datos introducidos por el usuario son válidos.

Se muestra un ejemplo, en este caso de comprobación de si un campo introducido por el usuario es un código postal válido o no. No se ejemplifica aquí el resto de métodos utilizados ya hay una gran cantidad de ellos y son todos muy similares.

```
//Devuelve el valor numérico del Código Postal. Si no es válido devuelve -1
public static int isPostalCode(string text)
{
    if (text.Length == 5)
        return isInteger(text);
    else
        return -1;
}
```

Figura 122: Comprobación de Código Postal

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.9 INFORMES

Se utilizan procedimientos almacenados para la generación de todos los informes ya que es la manera más rápida para la ejecución teniendo ya almacenados físicamente estos en la base de datos.

Estos son todos los procedimientos almacenados existentes:

- Gasto total
- Informe activa
- Informe alquiler de equipos
- Informe de CO^2
- Informe de consumo
- Informe excesos de potencia
- Informe factura
- Informe máxima activa
- Cambiar potencia contratada
- Cambiar tipo tarifa
- Cambiar a TUR
- Alarma Facturas
- Alarma ConsAct
- Alarma ConsReact
- Informe suministros de Socorro
- Informe control de facturas

Sólo se van a explicar los nuevos, ya que los otros ya fueron creados [1] y sólo se ha realizado alguna pequeña modificación sobre estos, luego prácticamente son los mismos.

4.9.1 INFORME DE SUMINISTRO DE SOCORRO

Procedimiento	Suministro de Socorro
Entradas	Fecha de Inicio y fecha fin para realizar la búsqueda (obligatorios) Posibilidad de buscar por niveles de jerarquía, potencia contratada, código de tarifa, y dirección del CUPS
Salida	Listado de CUPS que son suministros de socorro con los campos: Tarifa, CUPS, dirección, niveles de jerarquía, contrato, potencia, tensión…

Tabla 66: Entradas y Salidas del Informe Suministro de Socorro

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Figura 123: Informe Suministros Socorro

4.9.2 INFORME CONTROL DE FACTURAS

Procedimiento	Control de Facturas
Entradas	Fecha de Inicio y fecha fin para realizar la búsqueda (obligatorios) Posibilidad de buscar por CUPS
Salida	Gráfica con la representación en rojo de aquellos meses en que hay solapamientos en las fechas de las facturas, en amarillo si hay huecos y en verde si es correcto. Breve explicación por escrito de la información representada en el gráfico.

Tabla 67: Entradas y Salidas del Informe Control Facturas

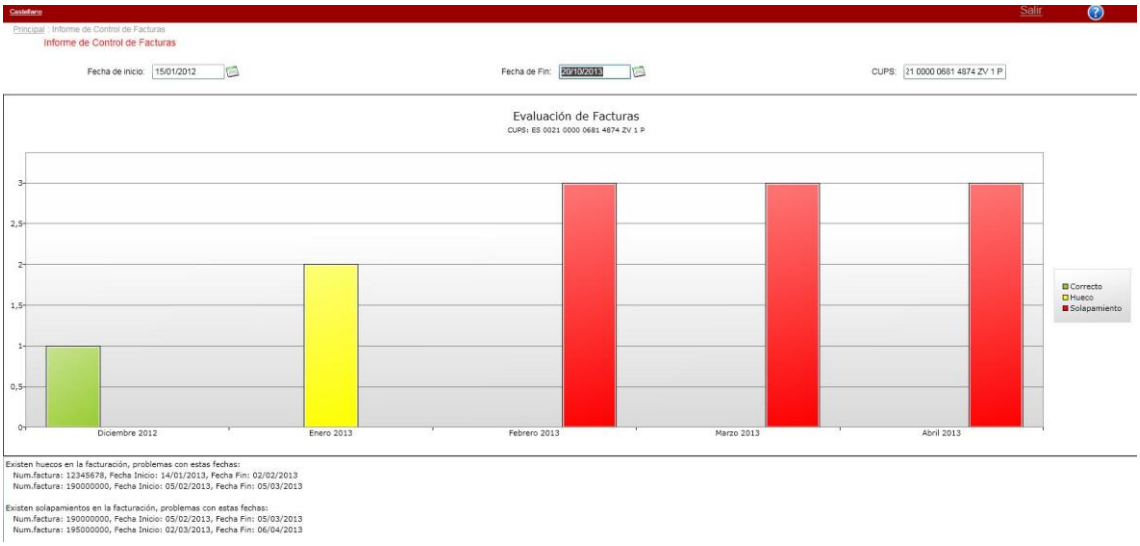


Figura 124: Informe Control Facturas

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.10 BACKUP BASE DE DATOS

En toda aplicación en la que exista una base de datos para almacenar información se requiere de la posibilidad de realizar una copia de seguridad de esta, ya que pueden pasar una serie de acontecimientos como:

- Catástrofe informática
- Se hayan borrado ciertos archivos accidentalmente (en este caso no pasa ya que no se permite al usuario borrar nada), existan archivos corruptos.

En principio se decidió por la generación de una copia de seguridad de manera automática cada cierto tiempo, sin embargo, no es lo más adecuado, ya que puede haber momentos que se introduzcan una gran cantidad de datos y sea necesaria esa copia.

Finalmente se optó por realizar la copia bajo demanda, es decir, el administrador será el único que se encargará de generar dicha copia de la manera más sencilla, sólo clicando un botón.

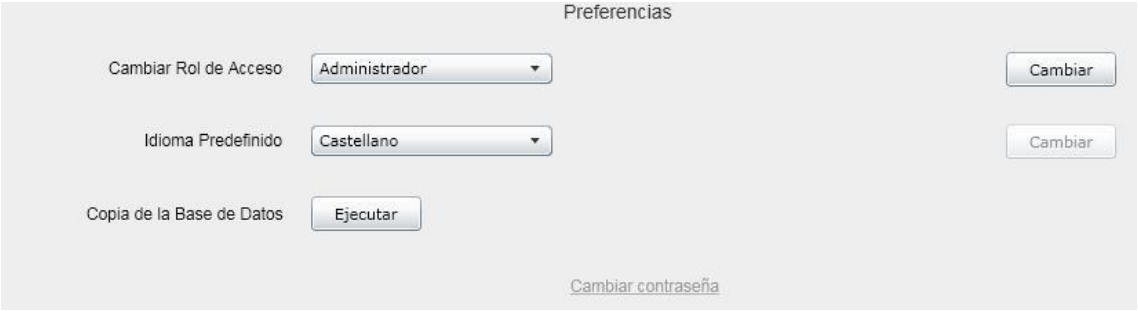


Figura 125: Botón de Crear la Copia de la Base de Datos

La copia de la base de datos se creará en una carpeta llamada Backups dentro del proyecto para que en el caso de que se necesite restaurar por cualquier motivo, un administrador experto lo pueda hacer sin problemas en cualquier momento.

4.11 ACTUALIZACIÓN PRECIOS DE LAS TARIFAS

Hablando con el cliente llegamos a la conclusión de que se necesitaba un lugar en la aplicación donde se pudiera actualizar de manera sencilla los precios de las tarifas.

También nos proporcionó más información sobre los precios. El precio total por cada periodo de la energía activa realmente se divide en dos partes, resultando ser la suma de ambas partes:

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- Precio de la energía activa: es el precio que sale a concurso y que se mantiene constante durante todo un año, sólo se modifica con la entrada del año.
- Peajes de acceso: están regulados por el gobierno y se suelen modificar cada 3 meses aprox.

Para poder realizar todos estos cambios se ha tenido que realizar una modificación en la base de datos añadiendo algún campo.

En el interfaz primero el usuario debe seleccionar la tarifa que quiere modificar y luego sólo permite introducir el valor de cada parte del precio, calculándose de manera automática el total y así evitarnos errores, ya que es la suma de ambos precios.

Además de los precios de las tarifas se suelen modificar al mismo tiempo los precios de la potencia.

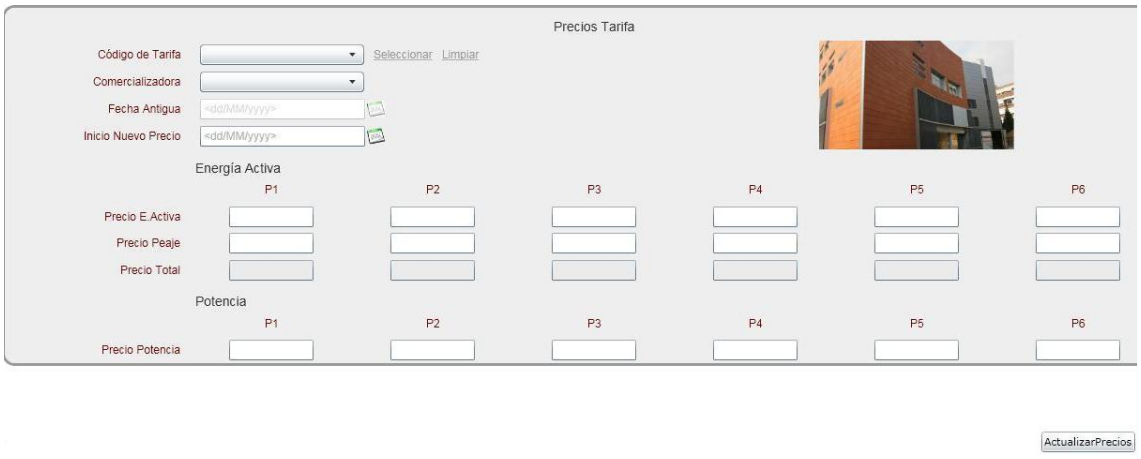


Figura 126: Pantalla de Actualización de Precios de las Tarifas

4.12 GRÁFICA EVOLUCIÓN PRECIOS TARIFA

Una funcionalidad que ha resultado interesante de implementar ha sido que el usuario pueda consultar en todo momento qué evolución han sufrido los precios de cada periodo de una tarifa determinada con el tiempo.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Cuando el usuario elije una tarifa al actualizar un precio de una tarifa aparece en la página un icono con la evolución de la tarifa.

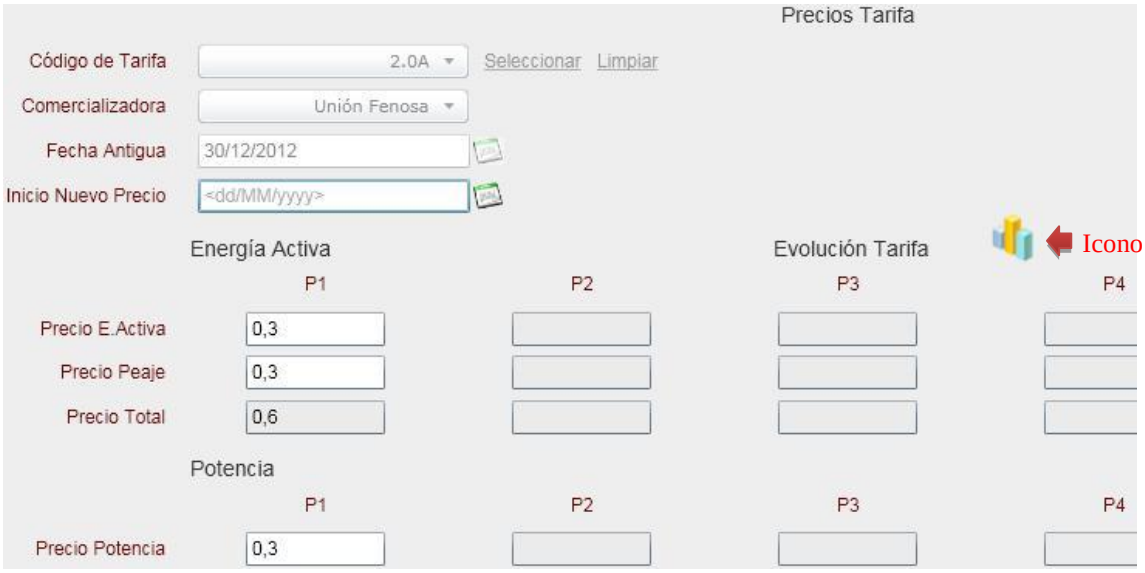


Figura 127: Icono que generará la gráfica

Pinchando en él, se abre una página desarrollada en ASP.NET, al igual que las páginas de inserción de facturas.

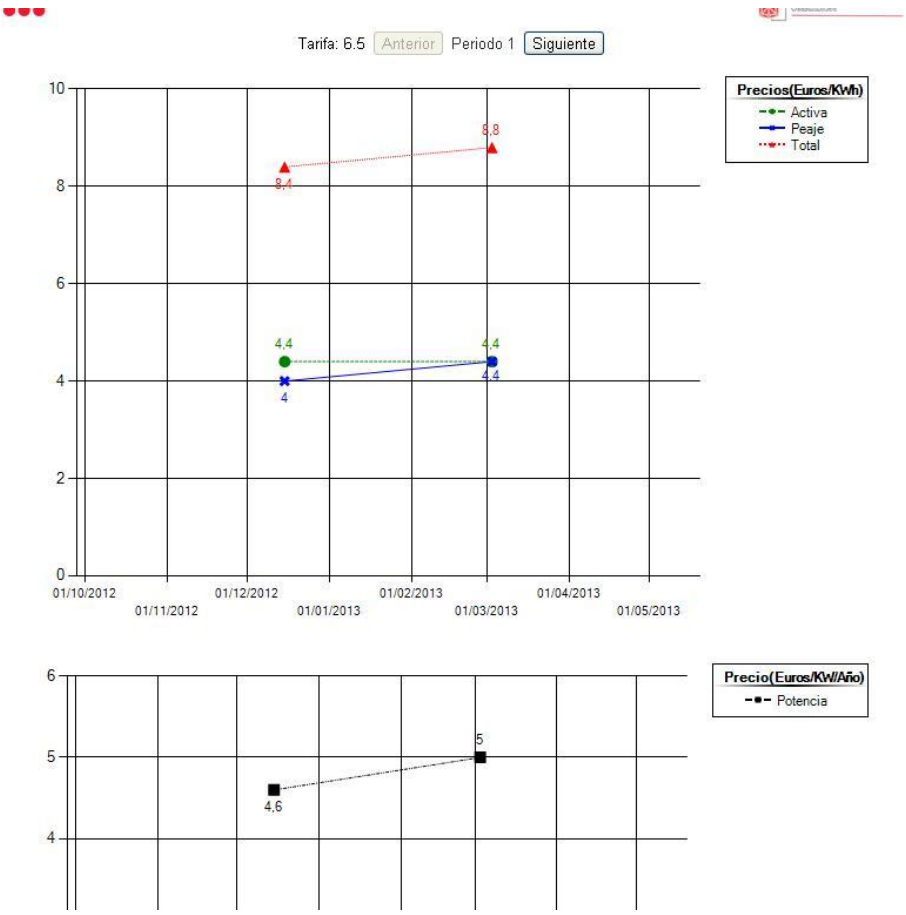


Figura 128: Gráfica de la Evolución de Precios

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

En la imagen se puede apreciar que existen dos gráficas, la de más arriba muestra la evolución de los precios de las energías activas, mostrando los precios tanto de la energía activa, como del peaje regulado por el gobierno, como del precio total de la energía activa y en la de abajo la evolución de los precios de la potencia.

Tanto en una como en otra, en el eje de abscisas se muestran las fechas y en el eje de coordenadas los precios en euros.

Además se puede consultar el valor de los precios de cada periodo, pudiéndose desplazar desde el periodo 1 hasta el periodo 6, pinchando en los botones anterior y siguiente.

En este caso no se ha requerido utilizar una política de seguridad ya que sólo se podrá acceder al gráfico a través de la página de actualización de precios de tarifas, y esta sólo es accesible si el usuario es administrador. Además de que se trata de mostrar información de consulta, no de inserción, o modificación de la base de datos.

4.13 CAMBIOS PARA FACILITAR EL USO AL USUARIO

Durante el desarrollo del proyecto se ha creído conveniente realizar una serie de cambios del interfaz del proyecto, ya sea por petición del cliente, por facilitarle tareas, por reducción a la hora de cometer errores, etc.

4.13.1 INSERCIÓN Y EDICIÓN DE EMPRESAS

Se ha cambiado la estructura del diseño de la página dividiendo todos los datos que anteriormente se introducían juntos ahora se separan en dos tipos, los datos de la empresa y los datos de la persona de contacto.

Además algunos datos a introducir pasan de ser obligatorios a no obligatorios, ya que no son siempre necesarios.

Antes:

Figura 130: Inserción Empresa Después

Después:

Figura 129: Inserción Empresa Antes

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.13.2 UNIDADES DE MEDIDA

Para facilitar la comprensión del usuario y que sepa en qué unidades hay que introducir cada dato, se facilita una etiqueta a la derecha de cada dato a insertar o editar con la unidad de medida de dicho dato.

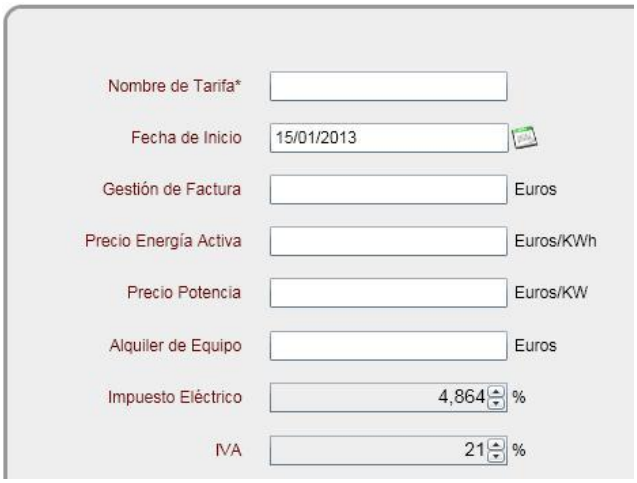


Figura 131: Unidades de Medida Añadidas

4.13.3 DESPLEGABLES VS CUADROS DE TEXTO

Debido a la existencia de datos que son limitados (existen una cantidad fija), se ha decidido cambiar algún bloque de texto por desplegables con los posibles valores a elegir.

Así se le facilita al usuario tanto a la hora de escritura como a la hora de cometer errores.

Antes:

Figura 132: Inserción Tarifas Después

Después:

Figura 133: Inserción Tarifas Antes

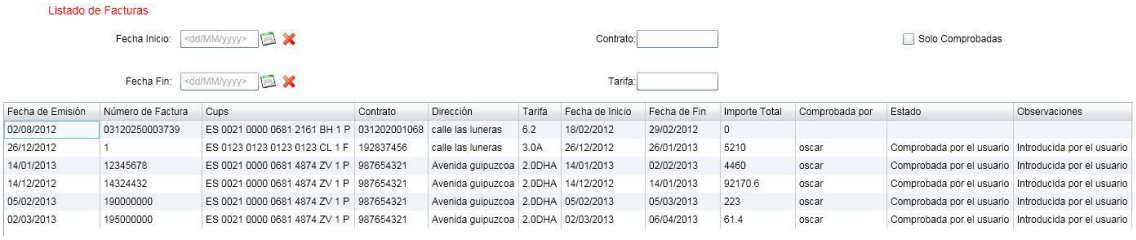
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.13.4 CONSULTA FACTURAS

Los campos a mostrar en la consulta de facturas han sido modificados ya que al cliente le quedaba más claro tal y como a él le proporcionan los informes de las facturas.

Además de añadir la opción de filtrar las facturas por:

- Fecha fin de factura
- Tarifa



Fecha de Emisión	Número de Factura	Cups	Contrato	Dirección	Tarifa	Fecha de Inicio	Fecha de Fin	Importe Total	Comprobada por	Estado	Observaciones
02/08/2012	03120250003739	ES 0021 0000 0681 2161 BH 1 P	031202001068	calle las luneras	6.2	18/02/2012	29/02/2012	0			
26/12/2012	1	ES 0123 0123 0123 0123 CL 1 F	192837456	calle las luneras	3.0A	26/12/2012	26/01/2013	5210	oscar	Comprobada por el usuario	Introducida por el usuario
14/01/2013	12345678	ES 0021 0000 0681 4874 ZV 1 P	987654321	Avenida guipuzcoa	2.0DHA	14/01/2013	02/02/2013	4460	oscar	Comprobada por el usuario	Introducida por el usuario
14/12/2012	14324432	ES 0021 0000 0681 4874 ZV 1 P	987654321	Avenida guipuzcoa	2.0DHA	14/12/2012	14/01/2013	92170.6	oscar	Comprobada por el usuario	Introducida por el usuario
05/02/2013	190000000	ES 0021 0000 0681 4874 ZV 1 P	987654321	Avenida guipuzcoa	2.0DHA	05/02/2013	05/03/2013	223	oscar	Comprobada por el usuario	Introducida por el usuario
02/03/2013	195000000	ES 0021 0000 0681 4874 ZV 1 P	987654321	Avenida guipuzcoa	2.0DHA	02/03/2013	06/04/2013	61.4	oscar	Comprobada por el usuario	Introducida por el usuario

Figura 134: Consulta Facturas

4.13.5 IVA E IMPUESTO ELÉCTRICO AUTOMÁTICO

Debido a todos los recortes de la crisis de 2012, el IVA ha sido afectado incrementándose desde el 18% al 21%, a partir del 9 de Septiembre de 2012.

El impuesto eléctrico hasta ahora no ha cambiado, pero en un futuro cabe la posibilidad de que se modifique.

Así que se ha desarrollado una parte de código que chequea si la fecha seleccionada por el usuario es inferior o superior al 9 de Septiembre de 2012 y según el caso modificar el IVA al 18% o al 21%.

```
//Cambiar el valor del IVA y del Impuesto Eléctrico al cambiar la fecha
private void InitDatePicker_SelectedDateChanged(object sender, SelectionChangedEventArgs e)
{
    //Al editar Tarifa está deshabilitado, sólo tiene sentido aquí en insertar
    //Fecha de corte IVA el 1 de Septiembre de 2012
    DateTime cutoffdate = DateTime.Parse("01/09/2012");
    if (InitDatePicker.SelectedDate != null)
    {
        //Cogemos la fecha seleccionada
        DateTime initDate = (DateTime)InitDatePicker.SelectedDate;
        if (initDate.CompareTo(cutoffdate) < 0)
            IVABox.Value = 18;
        else //Si es mayor o igual que ese dia al 21%
            IVABox.Value = 21;
```

Figura 135: Cambio de IVA

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.13.6 CARACTERES DE CONTROL DEL CUPS

La estructura de un código CUPS posee dos caracteres que se calculan a partir de los 16 numéricos separados en cuatro bloques, y sirven para la detección de errores.

Por ello, para evitar que el usuario cometa un fallo en la escritura de un código, se ha desarrollado el algoritmo que calcula estos dos caracteres.

```
//Calcular el CR (devuelve el CR si ha conseguido, -1 si no ha conseguido calcular)
public static string[] calculaCR(TextBox CUPSBox2, TextBox CUPSBox3, TextBox CUPSBox4, TextBox CUPSBox5)
{
    string[] cr = new string[2];
    Int64 numero = ConvInt64(fourTextBoxToCUPS(CUPSBox2, CUPSBox3, CUPSBox4, CUPSBox5));
    if (numero != -1)
    {
        //Los 16 dígitos módulo 529 y sale el resto. Con ese resto la C resto entre 23 y te quedas con el cociente(parte entera, ya te lo hace al convertir
        Int64 cociente = (numero % 529) / 23;
        Int64 resto = (numero % 529) % 23;

        cr[0] = map[cociente];
        cr[1] = map[resto];
    }
    else
    {
        cr[0] = "-1";
        cr[1] = "-1";
    }
    return cr;

}
```

Figura 136: Calcular el CR

En el interfaz no se le permite al usuario introducir estos caracteres del CUPS, así cuando introduzca el código del CUPS estos caracteres son calculados automáticamente:

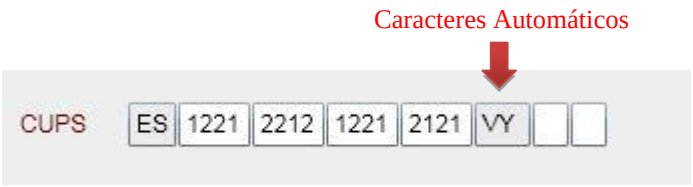


Figura 137: Cálculo Automático del CR

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.13.7 SUMINISTROS DE SOCORRO

Un CUPS que sea suministro de socorro ya se conoce de antemano si su fin es este o no. Por ello se ha añadido en el interfaz la posibilidad al insertar o editar un CUPS de marcarlo, como tal.

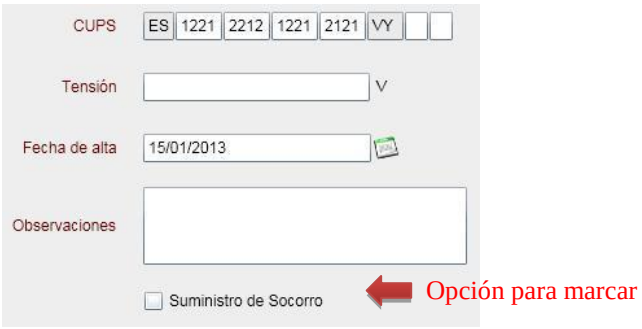


Figura 138

4.14 PROBLEMA BÚSQUEDA PRECIOS TARIFA NO VIGENTE

Tal y como se diseñó la base de datos era muy sencillo realizar una consulta del precio de la tarifa que estaba vigente, ya que en la tabla donde se asocia esto, el valor de la fecha de fin es null (indicando que es una asociación vigente en el sistema), sin embargo no era posible distinguir los precios de una tarifa u otra que no fuera la vigente, ya que el valor de la fecha fin es la de la fecha y hora exacta de la asociación o de la fecha y hora al darse de baja, y por lo tanto no hay una asociación entre las tarifas antiguas y sus precios.

El problema surgió cuando nos dispusimos a crear la consulta de la evolución de los precios de las tarifas en el tiempo, y también a la hora de crear la consulta de chequeo del precio de una tarifa durante la carga de las facturas a partir de un Excel.

En ambas se necesitaba saber qué precio tiene cada tarifa, sea la vigente o no.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Figura 139: Base de Datos Antes

Por ello se ha necesitado añadir una relación que asocie los precios con la tarifa a la que pertenece. Además de modificar alguna consulta SQL en el código para adaptarse a los nuevos cambios.

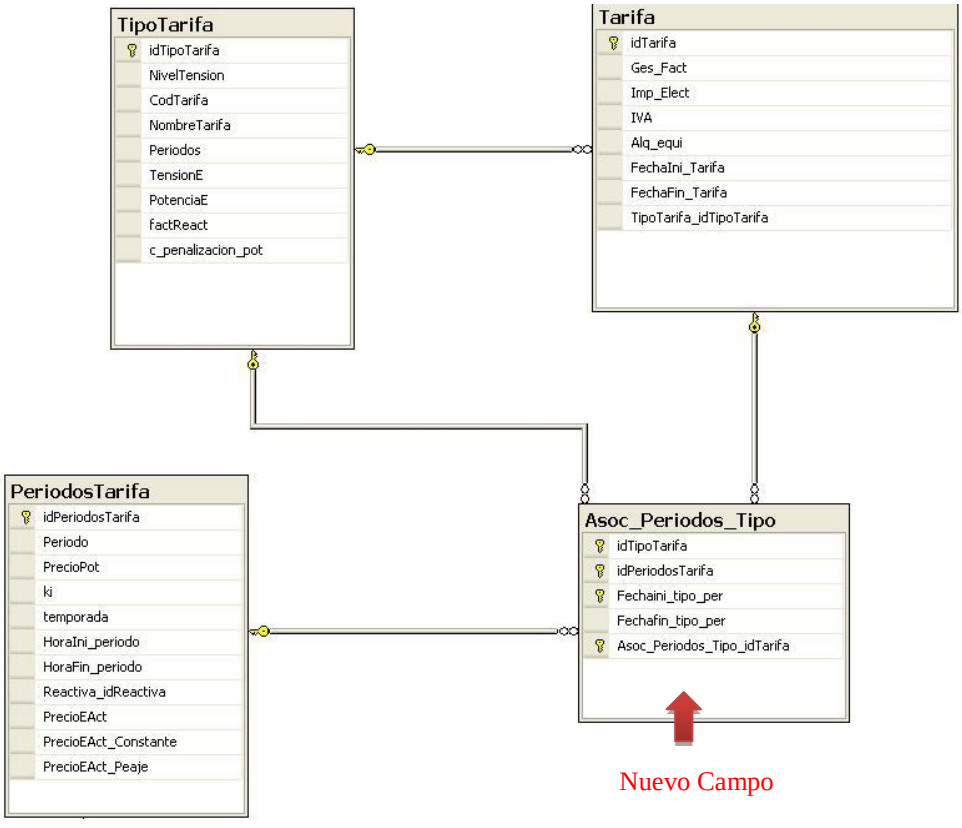


Figura 140: Base de Datos Después

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.15 LIBRERÍA EXCEL PARA IIS

Lo primero para poder manipular el Excel que Unión Fenosa proporciona fue hacer uso de la librería Microsoft.Office.Interop.Excel.

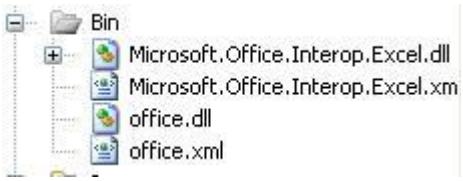


Figura 141: Librería Excel

Todo funcionaba correctamente en el entorno de pruebas, sin embargo al migrar la aplicación al servidor IIS comenzaron problemas. No se permitía el uso de ninguna función, como por ejemplo las de apertura el Excel, escritura, guardado, etc.

Buscando información en la Web vimos que el uso de esta librería en un servidor podía causar problemas:

- Los desarrolladores pueden utilizar la automatización de Microsoft Office para crear soluciones personalizadas que utilicen la capacidad y las funciones integradas en los productos de Office. Aunque este tipo de desarrollos mediante programación se pueden implementar en un sistema cliente con relativa facilidad, pueden surgir diversas complicaciones si la automatización tiene lugar con código del lado servidor como Páginas Active Server de Microsoft (ASP), DCOM o un servicio de Windows NT.

- Actualmente Microsoft no recomienda ni proporciona soporte técnico para la automatización de las aplicaciones de Microsoft Office desde una aplicación o componente cliente desatendido no interactivo (como ASP, DCOM y servicios de NT), ya que Office puede mostrar un comportamiento inestable y quedar interrumpido al ejecutarse en este entorno.

- Si decide utilizar una aplicación de Office en una solución de servidor, encontrará que carece de muchas de las capacidades necesarias para funcionar correctamente y estará arriesgando la estabilidad de toda la solución.

Debido a todo esto, se pasó a pensar alternativas. La primera era la ejecución remota desde el servidor de código que estuviera en el cliente con el fin de saltarnos el encapsulamiento de seguridad que el servidor tiene. Sin embargo los resultados fueron similares, ya que todo lo que ejecuta el servidor, primero lo chequea y pasa por el servidor antes de ejecutarse.

Finalmente, se optó por dejar de usar la librería y replantear otra forma de obtener la información del Excel. Por ello surgió da idea de obtener toda la información mediante

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

una conexión por OleDB, guardarla en un DataSet, y a partir de ahí crear un Excel XML manualmente con las etiquetas necesarias, y modificándolo a nuestro antojo.

```
//Crea el excel
public void CreateEXCEL(string nombre_ficheroExcel,string nombre_ficheroXML,fact3 facturas)
{
    //**********Generar un XML a partir de un Excel y mediante una conexión
    System.Data.OleDb.OleDbConnection MyConnection;
    System.Data.DataSet ds;
    System.Data.OleDb.OleDbDataAdapter MyCommand;
    int filas, columnas, columna_lec_ini_act_p1 = 0;
    StreamWriter sw = new StreamWriter(nombre_ficheroXML);

    //Consulta consiguiendo todos los datos (cabeceras y datos)
    MyConnection = new System.Data.OleDb.OleDbConnection("provider=Microsoft.Jet.OLEDB.4.0;Data Source=" + nombre_ficheroExcel + ";Extended Properties=" + "
    MyCommand = new System.Data.OleDb.OleDbDataAdapter("select * from [_DatosEmail$]", MyConnection);
    ds = new System.Data.DataSet();
    //Relleno en tabla llamada _DatosEmail
    MyCommand.Fill(ds, "_DatosEmail");
    //Filas y columnas(-1 ya que no incluye cabecera)
    filas = ds.Tables[0].Rows.Count + 1;
    columnas = ds.Tables[0].Columns.Count;
    //Nombres de las cabeceras
    DataColumnCollection drc = ds.Tables["_DatosEmail"].Columns;
    string[] cabeceras = new string[columnas];
    int v = 0;
    foreach (DataColumn dc in drc)
    {
        cabeceras[v] = dc.ColumnName;
        //Conseguir el número de columna(-1 en realidad), para pintar sum.socorro y estimadas
        if(dc.ColumnName=="LEC_INI_ACT_P1")
        {
            columna_lec_ini_act_p1 = v;
        }
        v++;
    }
    //ds.WriteXml(ruta + "excel2xml.xml");
    sw.WriteLine("<?xml version=\"1.0\"?>");
    sw.WriteLine("<?mso-application progid=\"Excel.Sheet\"?>");
    sw.WriteLine("<Workbook xmlns=\"urn:schemas-microsoft-com:office:spreadsheet\"");
    sw.WriteLine("xmlns:o=\"urn:schemas-microsoft-com:office:office\"");
    sw.WriteLine("xmlns:x=\"urn:schemas-microsoft-com:office:excel\"");
    sw.WriteLine("xmlns:ss=\"urn:schemas-microsoft-com:office:spreadsheet\"");
    sw.WriteLine("xmlns:html=\"http://www.w3.org/TR/REC-html40\">");
    //PROPIEDADES
    sw.WriteLine("<DocumentProperties xmlns=\"urn:schemas-microsoft-com:office:office\">");
    sw.WriteLine("<Created>" +
```

Figura 142: Creación del Excel

Esta solución fue la que resultó y la que finalmente se ha desarrollado.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.16 PROBLEMAS EXISTENTES

Actualmente en el proyecto hay ciertos aspectos de los que no existe suficiente información para poder implementarlo correctamente, o existen algunos problemas al querer almacenar un histórico de los cambios llevados a cabo en la base de datos del sistema.

4.16.1 TARIFAS TUR

El Servicio Navarro de Salud – Osasunbidea posee contratos con Unión Fenosa en los que una pequeña parte de CUPS hace uso de Tarifas de Último Recurso (TUR), ya que sólo se puede contratar una TUR para una potencia inferior a 10 Kw.

Por ello, las facturas proporcionadas hasta ahora han sido de tarifas que nos son TUR, por lo que no disponemos de información acerca de en qué formato o de qué manera nos proporcionan la información.

Por lo tanto, el programa de inserción de facturas a partir de un Excel está desarrollado para que luego, en el momento que se tenga mayor información se termine de implementar correctamente.

4.16.2 ALMACENAMIENTO DE HISTÓRICOS VS INCONSISTENCIA EN LA BASE DE DATOS

Inicialmente la aplicación se pensó para que guardara históricos de todos los cambios que se produzcan en la base de datos. Así poder llegar a saber en todo momento qué ha pasado, en qué hora, qué movimientos se han producido, etc.

El hecho de almacenar históricos en las tarifas tiene como objetivo el poder realizar cálculos en los informes con tarifas anteriores, sin embargo, no se ha barajado la posibilidad de que el usuario pueda cometer fallos al introducir las tarifas, por lo tanto, esa tarifa insertada erróneamente seguirá en la base de datos y se realizarán cálculos sobre ella incluyéndola, ya que no se distingue si es una tarifa correcta o incorrecta.

En el caso de que existan tarifas erróneas en la base de datos los informes generados puede que no reflejen resultados reales, sino datos erróneos.

También al poder cargar una factura que pertenece a una tarifa, ocurre el mismo problema, ya que si existen tarifas erróneas es posible asociar unas determinadas facturas a dichas tarifas. Y por lo tanto existirían inconsistencias.

Por ello como una tarea futura se propone el dar una vuelta de tuerca a este tema. Se pensó en la opción de crear otro usuario con privilegios superiores al administrador que

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

sea capaz de borrar datos, pero debe de estar muy seguro de lo que hace ya que el borrado de un dato correcto podría provocar una catástrofe en la base de datos.

4.17 PRUEBAS UNITARIAS

El sistema únicamente se trata de un prototipo, por ello no se puede hablar de pruebas de implantación. Lo que sí se ha hecho ha sido, para cada página de introducción de datos una prueba de caja negra para tener en cuenta los posibles errores del software. Se muestran aquí tres ejemplos:

4.17.1 INSERCIÓN DE CUPS

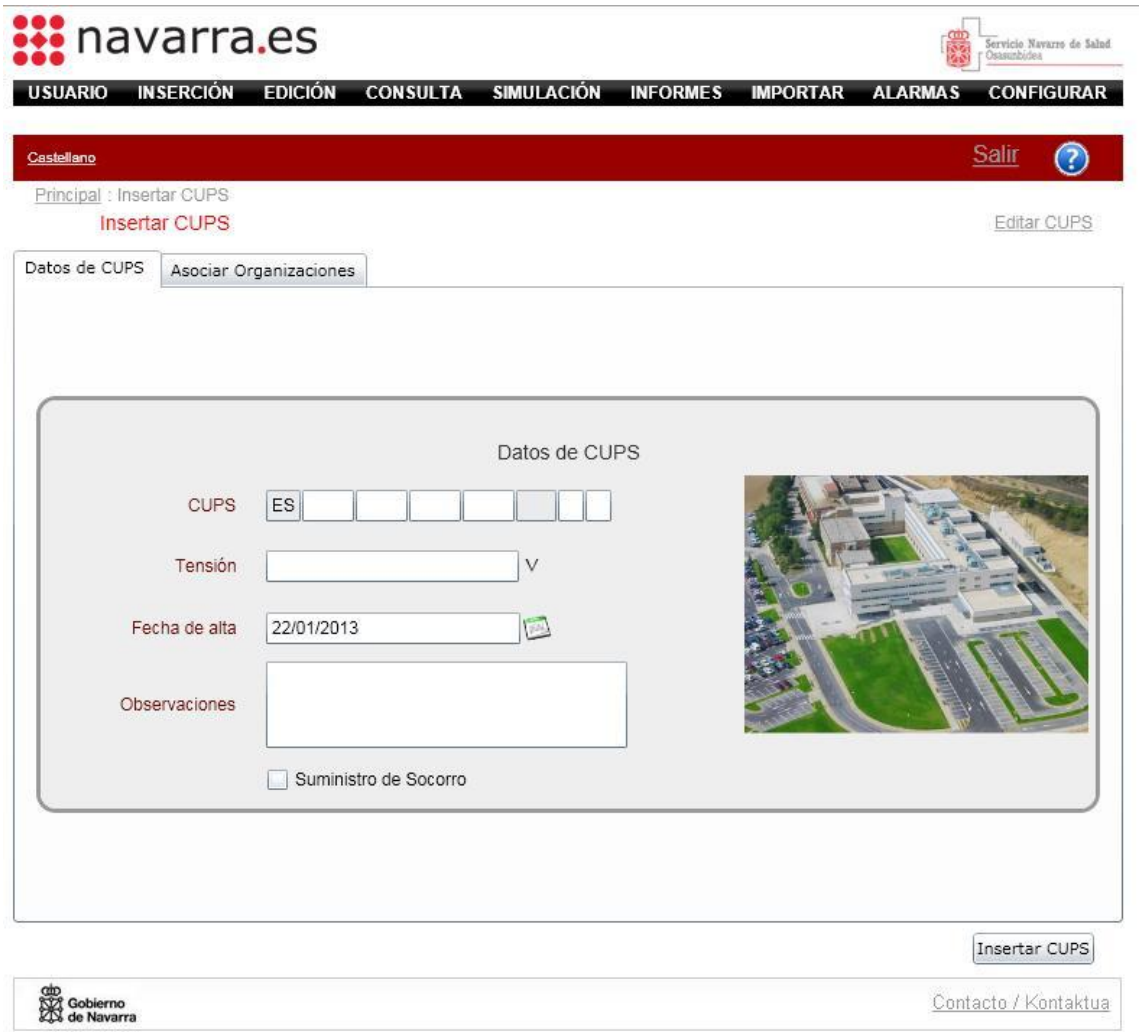


Figura 143: Prueba Inserción de CUPS

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Las clases de equivalencia con aquellos casos en los que la aplicación deberá funcionar correctamente y en los que deberá mostrar error son:

Clases de equivalencia válidas	*Clases de equivalencia no válidas*
1. Todos los campos de datos han sido introducidos	1. Falta algún campo de datos que no sea observaciones
2. Todos los campos de datos excepto observaciones han sido introducidos	2. El CUPS no tiene el formato correcto
3. El CUPS tiene el formato correcto	3. Ya existe un CUPS con ese código
4. No se ha indicado ninguna organización a asociar	
5. Se ha indicado alguna organización a asociar	

Tabla 68: Clases de equivalencia para la prueba de Inserción de CUPS

Se diseñan las siguientes pruebas:

1. Se introducen todos los campos de manera correcta y con un código CUPS no existente en el sistema, no asociando organizaciones.
2. Se introducen todos los campos de manera correcta y con un código CUPS no existente en el sistema, asociando una organización.
3. Se introducen todos los campos de manera correcta y con un código CUPS no existente en el sistema, asociando varias organizaciones.
4. Se introducen todos los campos de manera correcta excepto observaciones, y un código CUPS no existente en el sistema, sin asociar organizaciones.
5. Se introducen todos los campos de manera correcta excepto observaciones, y un código CUPS no existente en el sistema, asociando una organización.
6. Se introducen todos los campos de manera correcta excepto observaciones, y un código CUPS no existente en el sistema, asociando varias organizaciones.
7. No introducimos el código CUPS, pero sí el resto de datos.
8. Introducimos todos los datos pero el código CUPS no mantiene el formato correcto.
9. Introducimos todos los datos de manera correcta pero el código CUPS ya existe en el sistema el mismo que se introduce.
10. Introducimos todos los datos correctamente excepto el nivel de tensión.
11. Introducimos todos los datos correctamente, indicando que el CUPS es un suministro de socorro
12. Introducimos todos los datos de manera correcta pero se inserta un CUPS con los dos caracteres opcionales, y existe uno igual en la base de dato sin los caracteres opcionales introducidos.
13. Introducimos todos los datos de manera correcta pero se inserta un CUPS sin los caracteres opcionales, y existe uno igual en la base de datos con los caracteres opcionales introducidos.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Número de la prueba	*Resultado*
1	Inserción correcta
2	Inserción correcta
3	Inserción correcta
4	Inserción correcta
5	Inserción correcta
6	Inserción correcta
7	El sistema da error de falta de campos
8	El sistema da error de formato de CUPS
9	El sistema da error de CUPS ya existente
10	El sistema da error de falta de campos
11	Inserción correcta
12	El sistema da error de CUPS ya existente
13	El sistema da error de CUPS ya existente

Tabla 69: Resultado de la prueba de Inserción de CUPS

4.17.2 EDICIÓN DE CONTRATO

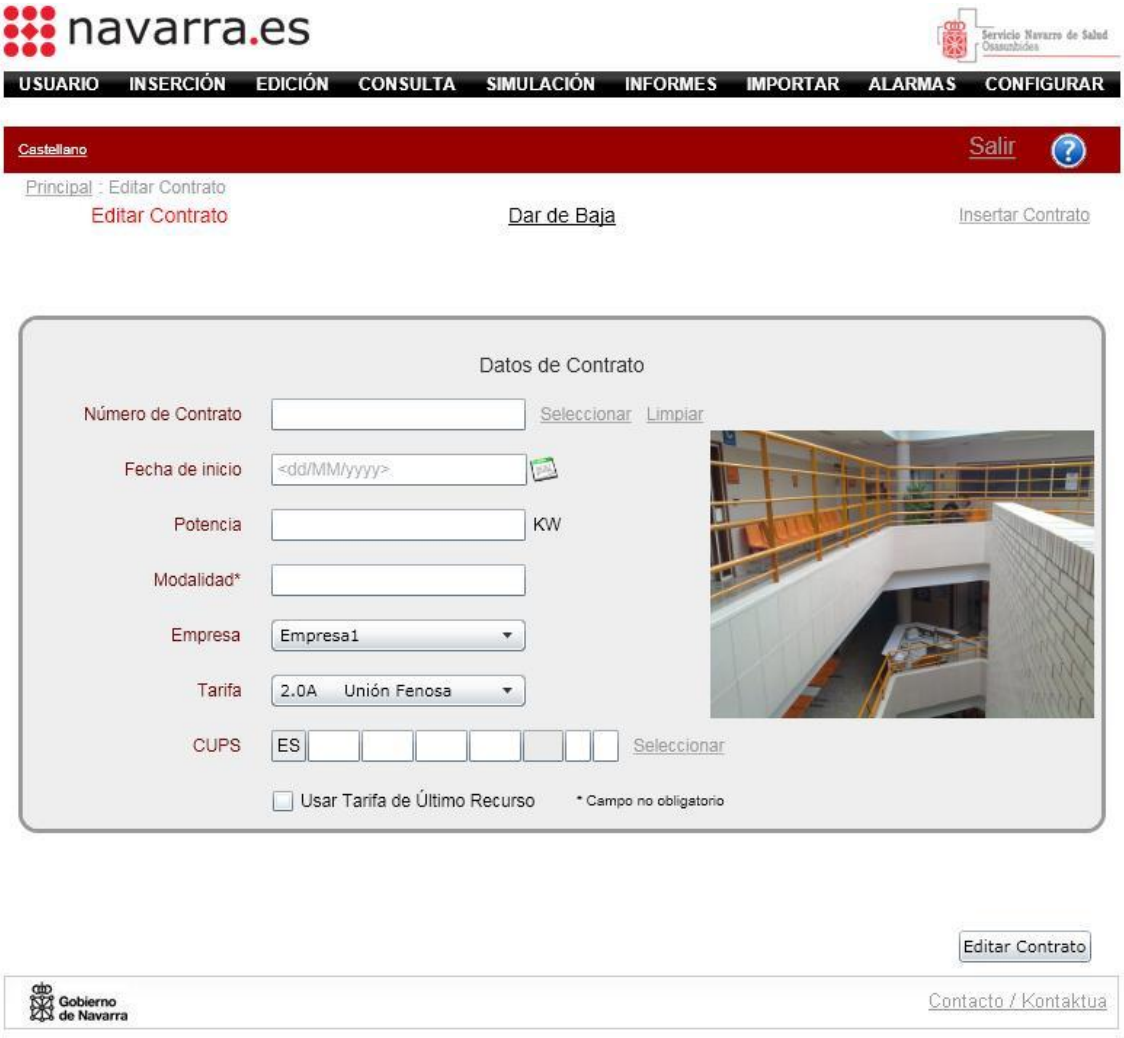


Figura 144: Prueba de Edición de Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Las clases de equivalencia con aquellos casos en los que la aplicación deberá funcionar correctamente y en los que deberá mostrar error son:

Clases de equivalencia válidas	*Clases de equivalencia no válidas*
1. Se ha indicado un contrato a modificar	1. No se introduce algún campo de datos
2. Se introducen todos los campos de datos	2. No se ha indicado tarifa ni TUR
3. Se introduce una fecha correcta	3. No se ha indicado empresa
4. Se ha indicado una tarifa	4. La tarifa no existe en el sistema
5. Se ha indicado una empresa	5. La empresa no existe en el sistema
6. La tarifa existe en el sistema	6. El CUPS no existe en el sistema
7. La empresa existe en el sistema	7. El CUPS no mantiene el formato correcto
8. No se ha indicado tarifa, pero sí que se use TUR	8. La potencia no es un valor numérico
9. Se ha indicado un CUPS ya existente.	9. Existe en el sistema otro contrato con el mismo número
10. El nuevo número de contrato ya existe en el sistema	
11. El CUPS tiene un formato correcto	

Tabla 70: Clases de equivalencia para la prueba de Edición de Contrato

Se diseñan las siguientes pruebas:

1. Se introducen todos los campos de manera correcta y con un código CUPS, empresa y tarifa existentes en el sistema, pero sin haber indicado el contrato a modificar.
2. Se introducen todos los campos de manera correcta y con un código CUPS, empresa y tarifa existentes en el sistema, habiendo indicado el contrato a modificar pero sin modificar ningún campo.
3. Se introducen todos los campos de manera correcta y con un código CUPS, empresa y tarifa existentes en el sistema, habiendo indicado el contrato a modificar modificando todos los campos.
4. Se introducen todos los campos de manera correcta, indicando que se va a usar una TUR.
5. Se introducen todos los campos pero el formato del CUPS no es el correcto.
6. Se introducen todos los campos pero el CUPS no existe en el sistema.
7. Se introducen todos los campos pero el formato de la fecha no es correcto.
8. Se introducen todos los campos de manera correcta, pero la potencia no es un número.
9. Se introducen todos los campos de manera correcta, pero el nuevo número de contrato ya existe en el sistema.
10. Se indican todos los campos correctamente excepto la empresa.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

11. Se indican todos los campos correctamente excepto la tarifa.
12. La empresa indicada no existe en el sistema.
13. La tarifa indicada no existe en el sistema.
14. No se indica CUPS.
15. No se indica alguno de los campos de datos.

Número de la prueba	*Resultado*
1	El sistema da error de contrato no seleccionado
2	Modificación correcta
3	Modificación correcta
4	Modificación correcta
5	El sistema da error de formato de CUPS
6	El sistema da error de CUPS inexistente
7	El sistema da error de fecha incorrecta
8	El sistema da error de valor de potencia
9	El sistema da error de número de contrato existente
10	El sistema da error de falta de campos
11	El sistema da error de falta de campos
12	No se puede llegar a esa situación
13	No se puede llegar a esa situación
14	El sistema da error de falta de campos
15	El sistema da error de falta de campos

Tabla 71: Resultado de la prueba de Edición de Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

4.17.3 ACTUALIZACIÓN DE LOS PRECIOS DE UNA TARIFA

Figura 146: Prueba Edición de Contrato

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Las clases de equivalencia con aquellos casos en los que la aplicación deberá funcionar correctamente y en los que deberá mostrar error son:

Clases de equivalencia válidas	*Clases de equivalencia no válidas*
1. Se ha seleccionado alguna tarifa existente	1. No se ha seleccionado una tarifa
2. Se ha introducido la fecha de inicio de la nueva tarifa	2. No se ha indicado la fecha de inicio de la nueva tarifa
3. La fecha introducida es correcta	3. La fecha introducida es incorrecta
4. La fecha introducida es posterior a la fecha de inicio de la antigua tarifa	4. La fecha de inicio de los precios de la nueva tarifa anterior a la antigua
5. Se ha modificado algún precio	5. La fecha de inicio de los precios de la nueva tarifa es igual a la antigua
6. Todos los precios son de tipo numérico	6. No se ha modificado ningún precio
	7. Existe algún precio que no es numérico

Tabla 72: Clases de equivalencia para la prueba de Actualización de Precios de Tarifa

Se han diseñado las siguientes pruebas:

1. No se selecciona ninguna tarifa a modificar
2. Se selecciona una tarifa pero no se inserta la fecha de inicio de la nueva tarifa
3. Se selecciona una tarifa y una fecha de inicio pero es anterior a la antigua fecha
4. Se selecciona una tarifa y una fecha de inicio pero es la misma que la antigua fecha
5. Se selecciona una tarifa, una fecha posterior y no se modifica ningún precio
6. Se selecciona una tarifa, una fecha posterior, pero se deja algún precio en blanco
7. Se selecciona una tarifa, una fecha posterior, y se modifica algún precio sin dejarse en blanco ninguno

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Número de la prueba	*Resultado*
1	El sistema da error de tarifa no seleccionada
2	El sistema da error de fecha no seleccionada
3	El sistema da error de que la fecha nueva debe ser posterior a la fecha antigua
4	El sistema da error de que la fecha nueva debe ser posterior a la fecha antigua
5	El sistema da error de que no se han cambiado los precios
6	El sistema da error de que faltan campos por rellenar
7	Actualización Correcta

Tabla 73: Resultado de la prueba de Actualización de Precios de Tarifa

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONCLUSIONES Y LÍNEAS FUTURAS

Capítulo final en el que se incluyen las conclusiones y posibles líneas futuras del proyecto desarrollado

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

5 CONCLUSIONES Y LÍNEAS FUTURAS

Con la finalización de este proyecto, se tiene un prototipo del programa, el cual ya está instalado en el portátil de un empleado del Servicio Navarro de Salud donde se continuarán realizando evaluaciones y pruebas al igual que han ido realizándose en estos últimos meses.

Como conclusión del proyecto desarrollado destacar sobre todo el trato con el cliente, lo cual es una experiencia que no se puede conseguir durante los años estudiados en la universidad y se agradece antes de entrar al mundo laboral, además de toda la programación desarrollada.

Ha sido una experiencia muy positiva para mí. Primero porque partí de un desarrollo ya implementado, el cual lo tienes que entender, adaptarte y continuar en la misma línea, lo cual no es nada fácil, además de que no se trata de un proyecto pequeño. Y segundo, el aprendizaje de las tecnologías Web que se usan hoy en día tales como las plataformas de desarrollo .NET y ASP, y también el aprendizaje del gestor de bases de datos SQL Server tanto su funcionamiento de forma independiente como integrado con el programa Visual Studio.

Recalcar que principalmente mi tarea ha consistido en arreglar bugs, adaptar una serie de funcionalidades a los cambios producidos e implementar nuevas funcionalidades.

Para terminar sólo queda indicar las líneas futuras que el proyecto podría tomar.

La primera sería una pequeña restructuración en la Base de Datos para poder adaptar la energía reactiva correctamente (ya que antes estaba pensada para cada tarifa, sin embargo no tiene sentido, tiene sentido para las facturas). Debería implementarse como una tabla independiente en la Base de Datos (no una tabla ligada a la tabla de precios de las tarifas) en la que se realizaría una carga inicial con los datos de los precios y los cosenos (o el usuario debe introducir), en donde únicamente se consultará el precio (a la hora de trabajar con las facturas) y en otra parte del programa permitiera modificarla en el caso de que algún precio variará (pero sólo la podría modificar un administrador).

La segunda se trata de una reflexión de si merece la pena o no crear un nuevo tipo de usuario (superadministrador) que pueda borrar registros de la Base de Datos. Esto es debido a que por motivos de seguridad y de almacenamiento de información para la generación de informes se creyó conveniente mantener un histórico de casi todo lo insertado (sea por error del usuario o por cambio en la realidad) y ahora existe algún problema, por ejemplo en el caso de que se produzcan errores por parte del usuario (algo habitual) a la hora de insertar o editar tarifas ya que existirán para siempre, y a la hora de cargar una factura puede que exista conflicto y se carguen relacionándose con una tarifa errónea. En el caso de que ocurra esto, también en la generación de algún informe puede que se muestren datos que no se corresponda con la realidad.

Una tercera tarea a implementar en el futuro sería que la aplicación diera la posibilidad de distribuir informes entre los usuarios. Es decir, que un usuario pudiera mandar uno o

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

varios informes a la "bandeja de entrada de otro usuario", y que el que reciba los pueda consultar, ya que hablando con el cliente comentó que podría ser útil y le facilitaría trabajo.

Otra tarea a añadir es la de controlar las facturas abonadas y las facturas estimadas. Según el cliente suele pasar que existan facturas cobradas erróneamente y debido a una reclamación del cliente (facturas con diferentes precios en las tarifas, otra potencia diferente de la contratada, etc.) o si la propia empresa comercializadora se ha dado cuenta, genera una nueva factura con los cambios correspondientes. Se necesitaría implementar un código que chequeara y tratara estos eventos. En cuanto a las facturas estimadas sería conveniente que se almacenaran de alguna manera en la base de datos y en el momento que llegara la factura real siguiente comprobar que el importe de diferencia entre una factura y otra se abone o se cobre correctamente.

También sería útil la generación una gráfica que mostrara de cada factura, de cada periodo, la potencia contratada contra la potencia consumida, así el usuario podría identificar de manera rápida si en existe algún contrato en el que reduciendo la potencia o incrementándola fuese capaz de ajustarla más al consumo habitual y así reducir en costos.

Otro tema que quedó pendiente con el cliente es la de incluir condensadores en los puntos de suministro que tienen un alto consumo de reactiva. Ya nos comentó algún precio de condensadores y que se amortizarían supuestamente en un periodo de tiempo pequeño, pero es un tema que hay que profundizar más y ver cómo controlar el consumo de reactiva para colocar o no condensadores.

Otra línea de desarrollo podría ser la inclusión de nuevos modelos de cálculo.
En la actualidad el Servicio Navarro de Salud – Osasunbidea tiene contratos con una compañía eléctrica Unión Fenosa y tuvo contratos hasta Marzo de 2012 con Endesa, sin embargo frente a la ausencia de datos de otras compañías el sistema sólo reconoce esos dos formatos de facturación.

También se tiene que tener en cuenta la traducción del idioma a euskara o incluso a algún otro. En esta fase del proyecto nos hemos dedicado más a implementar los requisitos funcionales y más importantes, por ello en un futuro existe la posibilidad de traducirlo tal y como se realizó en la primera fase del proyecto y tal y como en algún apartado anterior se ha especificado en más profundidad.

En cuantos a los informes que se generan en la aplicación, la mayoría tienen sentido con las facturas de Endesa y no con el tipo de facturación que hace Unión Fenosa, ya que esta última compañía proporciona diferente información que la primera compañía en referencia a sus facturas. Por ello, sería de gran utilidad la creación o generación de informes específicos que den información útil para tomar decisiones según la facturación que hace Unión Fenosa.

A última hora del proyecto surgió la idea crear otra aplicación independiente o agregar funcionalidad a esta para gestionar, no sólo las facturas eléctricas del Servicio Navarro de Salud – Osasunbidea tal y como se hace ahora mismo, sino también el gas.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

La última línea a comentar que resultaría interesante para su desarrollo en un futuro estaría constituida por la parte de alarmas. En la actualidad es el usuario el que introduce unos valores para las alarmas, que serán activadas si los consumos de un determinado CUPS exceden dichos umbrales.
Podría ser un desarrollo atractivo analizar de una manera más exhaustiva los consumos y generar un sistema con posibilidades de aprendizaje que activase las alarmas de forma automática si los valores se alejasen en mayor o menor medida del comportamiento habitual.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

BIBLIOGRAFÍA Y REFERENCIAS

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

6 BIBLIOGRAFÍA Y REFERENCIAS

[1] S. Huguet Pérez, "Desarrollo de una Herramienta Informática para la Gestión del Suministro Eléctrico", Proyecto Fin de Carrera, Universidad Pública de Navarra, 2011

[2] I. Trebol Araiz, "Herramienta Informática de Ayuda en la Decisión de Contratación de Tarifas eléctricas", Proyecto Fin de Carrera, Universidad Pública de Navarra, 2011

[3] API Online Microsoft SQL Server 2008 R2

http://msdn.microsoft.com/es-es/library/ms130214(v=sql.105).aspx

[4] API Online de Microsoft .NET C#

http://msdn.microsoft.com/es-es/library/67ef8sbd.aspx

[5] API Online Silverlight

http://msdn.microsoft.com/library/cc838158(VS.95).aspx

[6] Expresiones Regulares

http://msdn.microsoft.com/es-es/library/ms228595(v=vs.100).aspx

[7] Código de Estados de HTTP para IIS 7.0 y 7.5

http://support.microsoft.com/kb/943891

[8] Servidor IIS

http://www.canal-ayuda.org/a-informatica/serwebIIS.htm

[9] Permisos en SQL SERVER 2008 R2

http://msdn.microsoft.com/es-es/library/ms191291(v=sql.105).aspx

[10] Tipos de Tarifas Existentes

http://www.energylogic.es/tarifas%20electricas%20vigentes.pdf

[11] Estructura de un CUPS

http://noticias.juridicas.com/base_datos/Admin/res161109-itc.t5.html

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

[12] Facturas Estimadas

http://elpasatiempo.org/753/union-fenosa-o-como-metersela-doblada-a-la-estimada-clientela/

http://3tris3tigres.blogspot.com.es/2010/08/facturas-de-la-luz-estimadas.html

http://www.lavozdegalicia.es/noticia/economia/2012/01/20/recibo-luz-volvera-bimestral-basado-lecturas-reales/0003132705976508126917l.htm

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7 Anexo 1: ACTAS

En el presente anexo se adjuntan las actas obtenidas en cada reunión que se ha llevado a cabo entre el responsable del proyecto del SNS-O y los responsables de la realización del proyecto por parte de la Universidad Pública de Navarra.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.1 ACTA 1

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	24/7/2012
Hora	9.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- José Javier Astrain
- Óscar Echeverría

Orden del día

- Recapitulación del proyecto y resolución de los problemas que quedaron pendientes.

Temas tratados

- El IVA hasta el 1 de septiembre de 2012 será 18%, a partir de esa fecha 21%
- El impuesto eléctrico es regulado, se resolverá de la misma manera que el IVA, aunque sin fecha de cambio
- En los informes añadir la dirección y el nombre, falta que nos especifique qué datos desea que se muestren
- Colocar unidades a la derecha de los campos. Ej: Potencia (KW), Tensión (V), etc.
- Sobran algunos campos en la inserción de tarifas, falta especificar cuales exactamente.
- Idea de añadir el número de periodos como un desplegable de 1 a 6 para facilidad del usuario, y evitar posibles errores
- En las facturas ha cambiado el formato del documento de importación, ahora es en Excel, no como antes en XML

Conclusiones y decisiones

- Seguir resolviendo las incidencias, y continuar probando el programa a la busca de más incidencias

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.2 ACTA 2

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	28/8/2012
Hora	9.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- José Javier Astrain
- Óscar Echeverría

Orden del día

- Resolución de los problemas surgidos en la reunión anterior y surgimiento de nuevos problemas

Temas tratados

- La fecha límite de pago será 1 mes después de la fecha de inicio de factura
- Revisadas todas las etiquetas cambiadas
- Añadir desplegable en el nivel de tensión al insertar tarifa (alta, media y baja) y otro en el código de tarifa (2.0, 2.0A, 3.0,etc)
- Cambiar el nombre de tarifa por comercializadora y convertirlo en desplegable (Unión Fenosa, Iberdrola, Endesa, HC Energía y E.ON)
- Cambiar a no obligatorio el campo de potencia en las tarifas (se refiere al rango de potencia, el campo de potencia contratada está en el contrato)
- Cambiar a no obligatorio el campo modalidad
- Propuesta de generar una gráfica potencia demandada vs potencia consumida
- Propuesta de implementar un servicio de transmisión de informes para que se envíen entre usuarios los informes convenientes(mediante FTP)
- En el Excel identificar las facturas estimadas y los suministros de socorro
- Añadir en el programa de cargas de factura el chequeo del correcto cálculo del IVA y el IEE en el Excel, y que la potencia contratada realmente sea la que el Excel aparece
- Propuesta de habilitar una pantalla externa para introducir los nuevos incrementos energía PEA

Conclusiones y decisiones

- Seguir resolviendo las incidencias, y continuar probando el programa a la busca de más incidencias y ocurrencia de alguna nueva mejora a realizar
- Cuando haya una versión estable de la aplicación se instalará en el portátil de Óscar

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.3 ACTA 3

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	15/10/2012
Hora	9.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- José Javier Astrain
- Óscar Echeverría

Orden del día

- Resolución de los problemas surgidos en la reunión anterior y surgimiento de nuevos problemas

Temas tratados

- Dudas relacionadas sobre alguna columna del Excel a la hora de cargar facturas
- Los suministros de socorro "siempre" son suministros de socorro salvo acciones puntuales
- Idea de añadir al insertar un CUPS la opción de marcarlo como suministro de socorro
- Añadir un informe que muestre los suministros de socorro

Conclusiones y decisiones

- Disponiendo de una versión estable tanto de la aplicación general, como de la aplicación específica de inserción de las facturas desde un Excel se procederá a su instalación en el portátil de Óscar
- Seguir resolviendo las incidencias, y continuar probando el programa a la busca de más incidencias, y alguna nueva mejora a realizar.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.4 ACTA 4

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	26/10/2012
Hora	9.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- Óscar Echeverría

Orden del día

- Devolución del portátil (sólo con la instalación del programa general, no con el programa de las facturas, debido al surgimiento de problemas) y planteamiento de un par de dudas

Temas tratados

- Unión Fenosa proporcionará los Excel con sus facturas durante los meses de Agosto hasta final de año añadiendo alguna columna nueva
- Propuesta de dividir el precio de las tarifas en 2 partes, una parte fija que permanece constante durante el año entero y la otra que son los peajes que el gobierno sube cada 3 meses normalmente

Conclusiones y decisiones

- Seguir trabajando en solucionar el programa de las facturas y modificarlo para que funcione en el IIS
- Seguir resolviendo más problemas

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.5 ACTA 5

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	6/11/2012
Hora	9.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- Óscar Echeverría

Orden del día

- Instalación en el portátil el programa de inserción de facturas

Temas tratados

- Instalación del programa

Conclusiones y decisiones

- Oscar mirará los informes, los probará y verá qué columnas son necesarias o innecesarias para su posterior modificación
- Pensar en nuevos avances

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.6 ACTA 6

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	16/11/2012
Hora	12.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- Óscar Echeverría
- José Javier Astrain

Orden del día

- Seguir resolviendo dudas y surgimiento de nuevas necesidades

Temas tratados

- Añadir un botón de limpiar en insertar CUPS, contrato
- Cambiar etiquetas Añadir y Eliminar en vez de + y -
- Al actualizar precios de una tarifa: posibilidad también de actualizar los precios de la potencia
- Eliminar lo del coseno y el precio de la reactiva (Inhabilitar en principio), nos hemos dado cuenta que no tiene sentido poner para cada tarifa, se trata de una tabla prefijada
- Cambiar interfaz más amigable al insertar y editar una empresa
- Añadir en BD dos campos inicio consumo y fin de los contadores y posteriores cambios (chequeo solapamiento de contadores)
- Generar una gráfica que muestre la variación de los precios de las tarifas en el tiempo

Conclusiones y decisiones

- Seguir trabajando, continuar encontrando errores y posibles modificaciones, sobre todo los campos necesarios de las consultas e informes

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.7 ACTA 7

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	28/11/2012
Hora	10.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- Óscar Echeverría
- José Javier Astrain

Orden del día

- Seguir resolviendo dudas

Temas tratados

- Habilitar o deshabilitar una serie de parámetros de las tarifas en función de la compañía eléctrica seleccionada
- Realizar un sistema backup de la BD bajo demanda, cuando el usuario presione un botón

Conclusiones y decisiones

- Continuar trabajando

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.8 ACTA 8

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	21/12/2012
Hora	10.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- Óscar Echeverría
- José Javier Astrain

Orden del día

- Resolución de algún nuevo problema y muestreo de los cambios realizados

Temas tratados

- Mejorar la consulta de facturas añadiendo la columna de importe total, tarifa, dirección, y la reordenación de campos
- Además de añadir algún campo de filtrado de facturas: cups, tarifa y fecha de fin de tarifa
- Añadir un chequeo del nombre del nuevo periodo que sea "(string) (espacio) (número entre 1 y 6)", ya que se ha necesitado que sea de esa estructura
- Revisión de algún error surgido

Conclusiones y decisiones

- Resolver los errores surgidos e instalar el programa con la última actualización en el portátil

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

7.9 ACTA 9

Título del proyecto	Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica
Fecha	21/1/2013
Hora	9.00
Lugar	UPNA

Asistentes

- Óscar Araiztegui
- Óscar Echeverría

Orden del día

- Resolución de los últimos problemas o mejoras recopilados por el cliente durante este último mes y redactados en un informe

Temas tratados

- Cambiar y añadir una serie de etiquetas a algún campo
- Añadir algún campo en alguna consulta
- Los errores que aparecen por pantalla al insertar factura desde un Excel, que se vean también en el Excel

Conclusiones y decisiones

- Resolver las propuestas del cliente

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

8 Anexo 2: DIAGRAMAS

En el siguiente anexo se adjunta el diagrama de la base de datos que ya se encuentra en el documento pero en tamaño mayor con el fin de facilitar su visualización.

Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Backups

Column Name	Data Type
fecha	datetime

Festivo

Column Name	Data Type
idFestivo	int
diaFestivo	datetime

Seguridad

Column Name	Data Type
idUsuario	int
idRol	int
clave	int

Nivel1

Column Name	Data Type
idNivel1	int
Nombre	varchar(50)

Nivel2

Column Name	Data Type
idNivel2	int
Nombre	varchar(50)
DireccionPral	varchar(50)
CPPral	varchar(50)
LocalidadPral	varchar(50)

Nivel3

Column Name	Data Type
idNivel3	int
Nombre	varchar(50)
Direccion	varchar(50)
CP	varchar(50)
Localidad	varchar(50)
Tipo	varchar(50)

Nivel4

Column Name	Data Type
idNivel4	int
Nombre	varchar(50)

Asoc1_2

Column Name	Data Type
idNivel1	int
idNivel2	int
FechaIni_Asoc12	datetime
FechaFin_Asoc...	datetime

Asoc2_3

Column Name	Data Type
idNivel2	int
idNivel3	int
FechaIni_Asoc23	datetime
FechaFin_Asoc23	datetime

Asoc3_4

Column Name	Data Type
idNivel3	int
idNivel4	int
FechaIni_Asoc34	datetime
FechaFin_Asoc34	datetime
PorcentajeOcup	float

Asoc4_CUPS

Column Name	Data Type
idCUPS	int
idNivel4	int
FechaIni_Asoc4C	datetime
FechaFin_Asoc4c	datetime

FK_Asoc1_2_Nivel1
FK_Asoc1_2_Nivel2
FK_Asoc2_3_Nivel2
FK_Asoc2_3_Nivel3
FK_Asoc3_4_Nivel3
FK_Asoc3_4_Nivel4
FK_Asoc4_CUPS_Nivel4
FK_Asoc4_CUPS_CUPS

ConsAct_Aux

Column Name	Data Type
idActiva	int
idFactura	int
periodo	int
consumoAct	float
maximoAct	float
aciAct	float
lecIni	float
lecFin	float

ConsReact_Aux

Column Name	Data Type
idReactiva	int
idFactura	int
periodo	int
consumoReact	float

FK_ConsAct_Factura_Aux
FK_ConsReact_Factura_Aux

ConsAct

Column Name	Data Type
idActiva	int
idFactura	int
periodo	int
consumoAct	float
maximoAct	float
aciAct	float
lecIni	float
lecFin	float

Factura_Aux

Column Name	Data Type
idFactura	int
FechaIni_Factura	datetime
FechaFin_Factura	datetime
NumFactura	varchar(20)
FechaLimitePago	datetime
Contrato_idcontrato	int
FechaEmision	datetime

FK_Factura_Contrato_Aux

Registro

Column Name	Data Type
idFactura	int
idUsuario	int
fechahora	datetime
observaciones	varchar(50)
estado	varchar(50)

FK_ConsAct_Factura
FK_Registro_Factura
FK_ConsReact_Factura

Factura

Column Name	Data Type
idFactura	int
FechaIni_Factura	datetime
FechaFin_Factura	datetime
NumFactura	varchar(20)
FechaLimitePago	datetime
Contrato_idcontrato	int
FechaEmision	datetime

FK_Factura_Contrato

ConsReact

Column Name	Data Type
idReactiva	int
idFactura	int
periodo	int
consumoReact	float

Contrato

Column Name	Data Type
idContrato	int
FechaIni_Contrato	datetime
FechaFin_Contrato	datetime
Potencia	float
Modalidad	varchar(50)
CUPS_idCUPS	int
Empresa_idEmpresa	int
TUR_idTUR	int
Tarifa_idTarifa	int
numcontrato	varchar(25)

FK_Contrato_CUPS
FK_Contrato_Tarifa
FK_Contrato_TUR

CUPS

Column Name	Data Type
idCUPS	int
CUPS	varchar(50)
Tension	varchar(15)
FechaInstal_CUPS	datetime
FechaBaja_CUPS	datetime
Observaciones	varchar(250)
SumSocorro	int

FK_RegistrosConsumo_CUPS
FK_Contrato_Empresa
FK_UsuarioAlarma_CUPS
FK_Rol_Cups_CUPS
FK_UmbralesAlarma_CUPS

Alarma

Column Name	Data Type
FechaIni_Alarma	datetime
FechaFin_Alarma	datetime
TipoAlarma	nchar(10)
Descripcion	nvarchar(MAX)
idAlarma	int
UsuarioAlarma_idUsuarioAlarma	int

Rol_Cups

Column Name	Data Type
idRol	int
idCups	int
resp_alarma	bit

UsuarioAlarma

Column Name	Data Type
CUPS_idCUPS	int
Usuario_idUsuario	int
idUsuarioAlarma	int

FK_Alarma_UsuarioAlarma
FK_Rol_Cups_Rol
FK_UsuarioAlarma_Usuario

Rol

Column Name	Data Type
idRol	int
Descripcion	varchar(25...
Permiso	varchar(50)

Usuario

Column Name	Data Type
idUsuario	int
DirectorioActivo	varchar(50)
idioma	varchar(15)
usuario	varchar(50)
contrasena	varbinary(12...

FK_Usuario_Rol_Rol
FK_Usuario_Rol_Usuario

Usuario_Rol

Column Name	Data Type
idUsuario	int
idRol	int
activo	bit

FK_UmbralesAlarma_Usuario

UmbralesAlarma

Column Name	Data Type
duracion	float
umbralActiva	float
umbralReactiva	float
CUPS_idCUPS	int
Usuario_idUsuario	int
idUmbralesAlarma	int
email	varchar(50)

RegistrosConsumo

Column Name	Data Type
idRegistrosConsumo	int
FechaHora_Reg	datetime
E_Act	float
E_React	float
Real	bit
CUPS_idCUPS	int

TipoTarifa

Column Name	Data Type
idTipoTarifa	int
NivelTension	varchar(50)
CodTarifa	varchar(50)
NombreTarifa	varchar(50)
Periodos	int
TensionE	varchar(10)
PotenciaE	varchar(10)
factReact	bit
c_penalizacion_pot	float

Tarifa

Column Name	Data Type
idTarifa	int
Ges_Fact	varchar(50)
Imp_Elect	float
IVA	float
Alq_equi	varchar(50)
FechaIni_Tarifa	datetime
FechaFin_Tarifa	datetime
TipoTarifa_idTipoTarifa	int

FK_Tarifa_TipoTarifa

TUR

Column Name	Data Type
idTUR	int
FechaIni_TUR	datetime
FechaFin_TUR	datetime
Ges_Fact	varchar(50)
Imp_Elect	float
IVA	float
Alq_equi	varchar(50)
PrecioPotTUR	float
PrecioEactTUR	float
NombreTarifa	varchar(50)

FK_Asoc_Periodos_Tipo_TipoTarifa
FK_Asoc_Periodos_Tipo_idTarifa

PeriodosTarifa

Column Name	Data Type
idPeriodosTarifa	int
Periodo	varchar(1...
PrecioPot	float
ki	float
temporada	varchar(5)
HoraIni_periodo	time(7)
HoraFin_periodo	time(7)
Reactiva_idReactiva	int
PrecioEAct	float
PrecioEAct_Constante	float
PrecioEAct_Peaje	float

Asoc_Periodos_Tipo

Column Name	Data Type
idTipoTarifa	int
idPeriodosTarifa	int
Fechaini_tipo_per	datetime
Fechafin_tipo_per	datetime
Asoc_Periodos_Tipo_idTarifa	int

FK_Asoc_Periodos_Tipo_PeriodosTarifa
FK_PeriodosTarifa_Reactiva

Reactiva

Column Name	Data Type
coseno	varchar(15)
precioReactiva	float
idReactiva	int

Empresa

Column Name	Data Type
idEmpresa	int
CIF	varchar(10)
RazonSocial	varchar(50)
DirFiscal	varchar(50)
CNAE	varchar(50)
Nombre	varchar(50)
Direccion	varchar(50)
Email	varchar(50)
Telefono	varchar(15)
Fax	varchar(15)
PersonaContacto	varchar(15)
Movil	varchar(15)

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erreserbatuta dira

Desarrollo de una herramienta de gestión y supervisión de facturación eléctrica

Oscar Echeverría Esparza

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INTRODUCCIÓN

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

PROBLEMA A RESOLVER

- Servicio Navarro de Salud (SNS-O) formado de muchos edificios
- Muchas facturas, gran trabajo de facturación e informes
- Necesitan de una herramienta para:
  - Procesar facturas automáticamente
  - Genere informes
  - Realice simulaciones
  - Presente alertas
  - Gestión en cualquier lugar de forma sencilla

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ESTADO DEL ARTE

- En 2009 nueva ley de comercialización y distribución de la energía
- Multiplicación de las ofertas energéticas
- Formato de presentación de las facturas libre
- Estructura en constante cambio, es especial
- No existe herramienta que se adapte

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

OBJETIVOS

- Integración de la información relacionada con los contratos de suministro eléctrico de todos los centros del SNS-O
- Control centralizado y distribuido de la facturación
- Reducción del gasto a través del análisis de los consumos y las tarifas
- Trabajo en red, tanto en la introducción de datos como en la consulta y administración de la herramienta

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

SOLUCIÓN PROPUESTA

- Interfaz Web: Accesible desde cualquier lugar
- El SNS-O usa corporativamente tecnologías de Microsoft
- Ya existe el modelo de datos: PFC Desarrollado por Sandra Huguet Pérez en SQL Server 2008
- Ya existe el desarrollo de la interfaz web utilizando Microsoft Silverlight: PFC Desarrollado por Igor Trébol Araiz
- Continuar arreglando bugs , añadiendo nuevas funcionalidades que el SNS-O crea necesarias y adaptando cambios en el entorno

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

METODOLOGÍA

- Seguir con la misma metodología usada hasta el momento
- La metodología es: El Proceso Unificado de Desarrollo Software (RUP-UP)
  - Dirigido por Casos de Uso
  - Iterativo e incremental
- Elegida por:
  - La posibilidad de desarrollar la aplicación de forma iterativa y de adecuarse bien al proyecto
  - Y por ser una metodología conocida

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

REQUISITOS FUNCIONALES

- Estructura del sistema planteada por el SNS-O
- 3 Módulos
- Un tipo de usuario para cada módulo

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Módulo de Administración

- Usuario Administrador

- Gestión de Usuarios y Perfiles
- Configuración días festivos
- Creación de copia de seguridad de la BD

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Módulo de Datos

- Usuario de Datos

- Inserción en el sistema de todos los elementos:
  - Organizaciones
  - CUPS
  - Tarifas
  - Facturas
  - …
- Al final se decidió que es el usuario administrador el que introducirá los datos.
- El usuario de Datos solamente introduce y edita facturas.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Módulo de Consulta

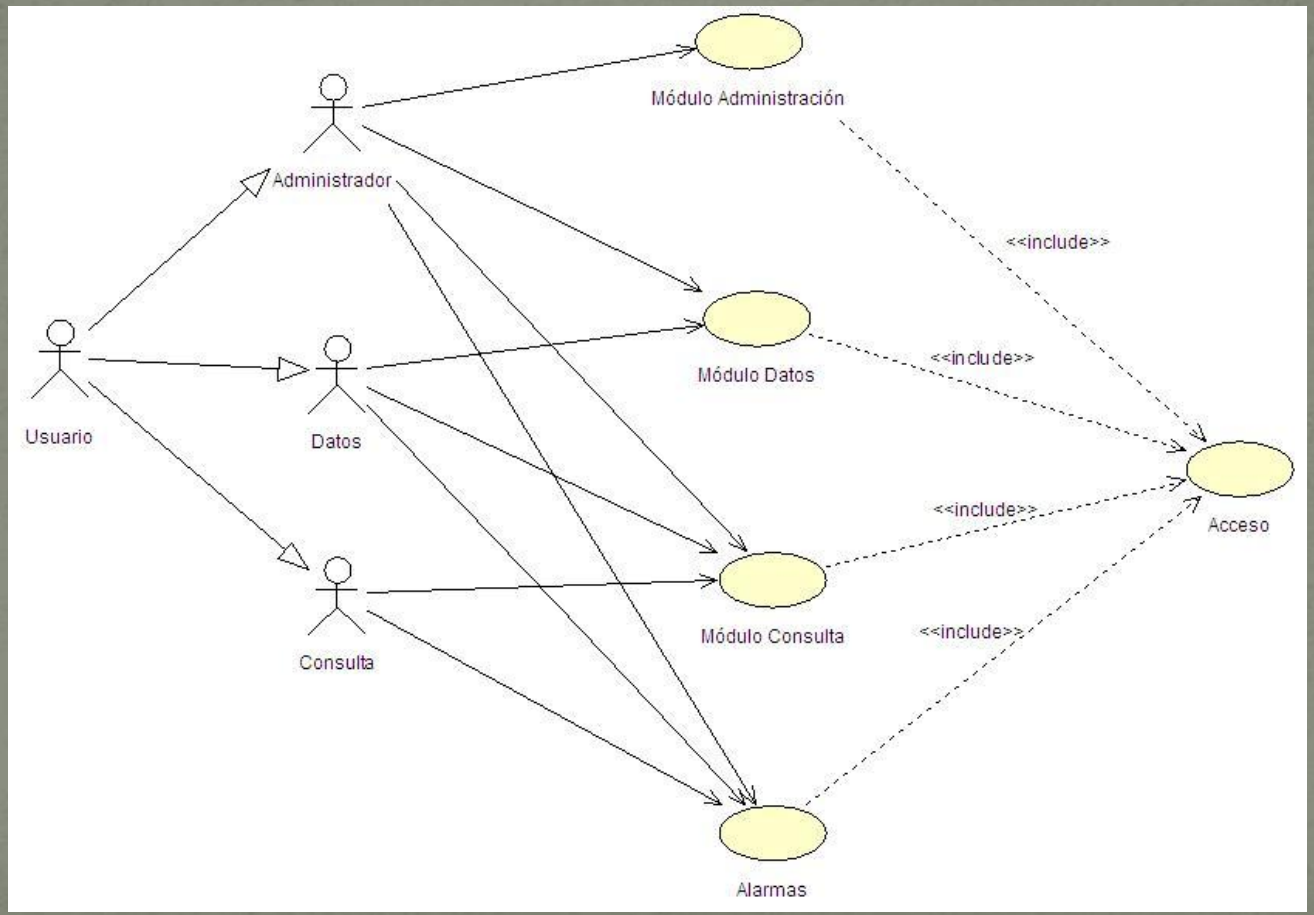
- Usuario de Consulta.

- Consultas o listados filtrables por uno o más campos de datos.
- Informes.
- Simulaciones.
- Comprobación de Facturas

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS DE LOS CASOS DE USO

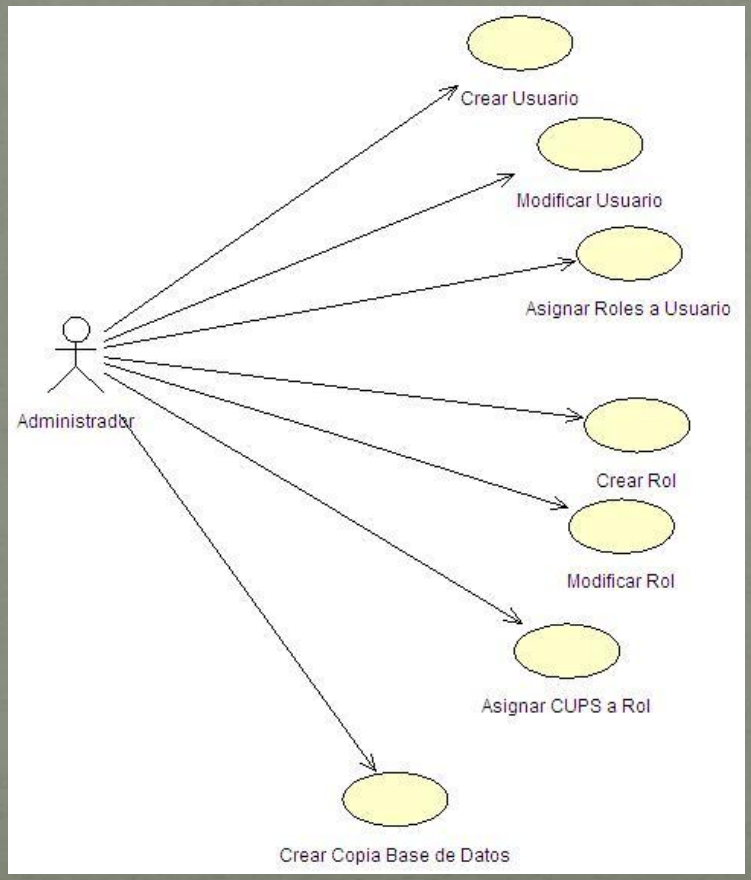
- CASO DE USO DEL SISTEMA



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS DE LOS CASOS DE USO

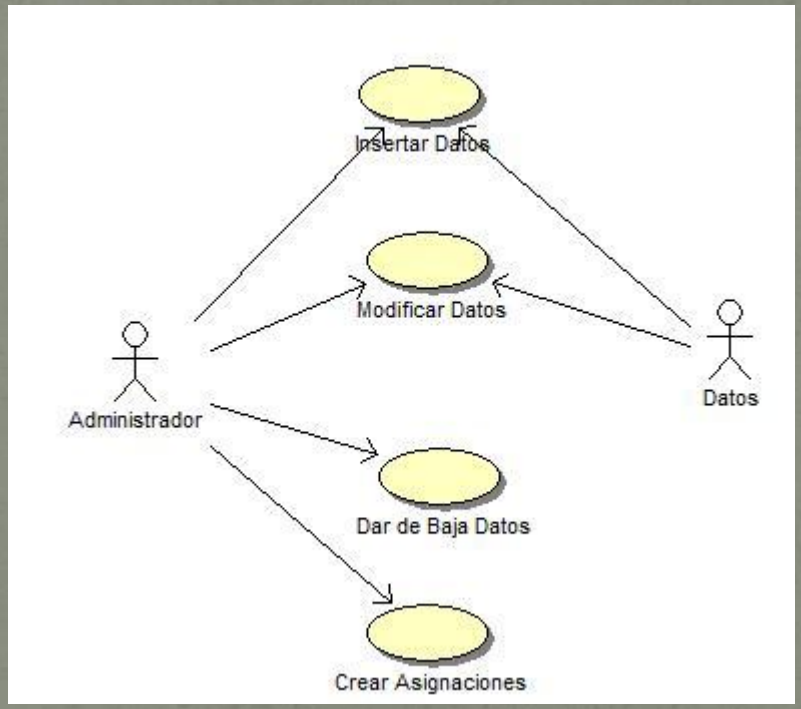
- Módulo de Administración



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS DE LOS CASOS DE USO

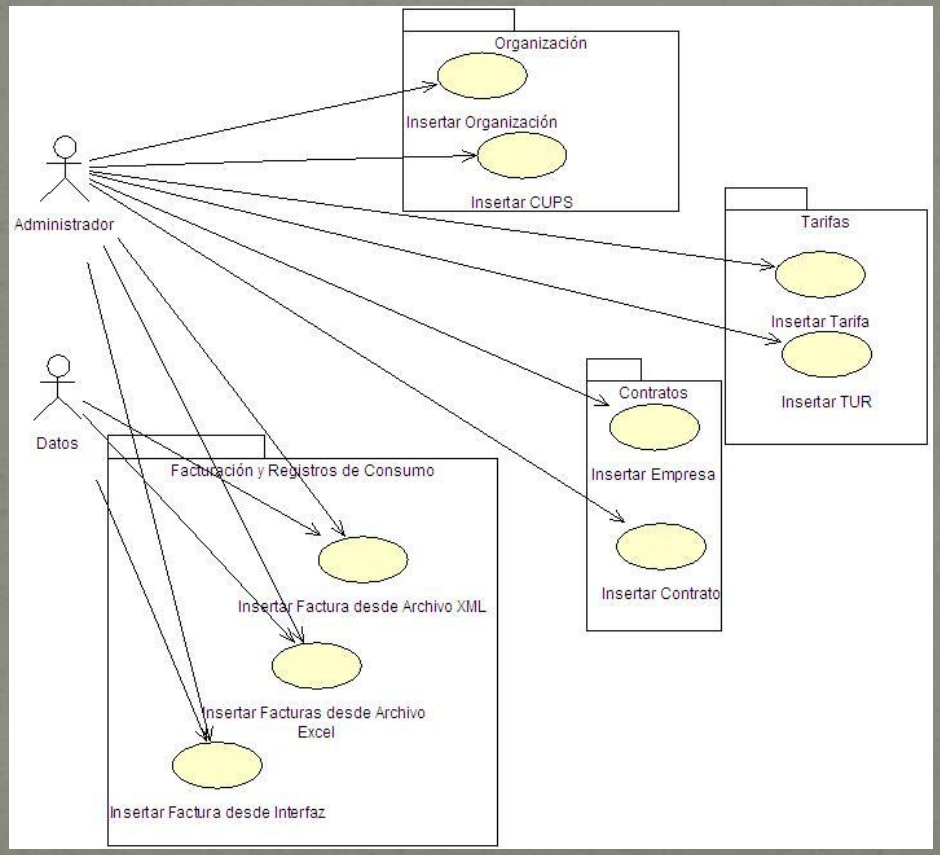
- Módulo de Datos



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS DE LOS CASOS DE USO

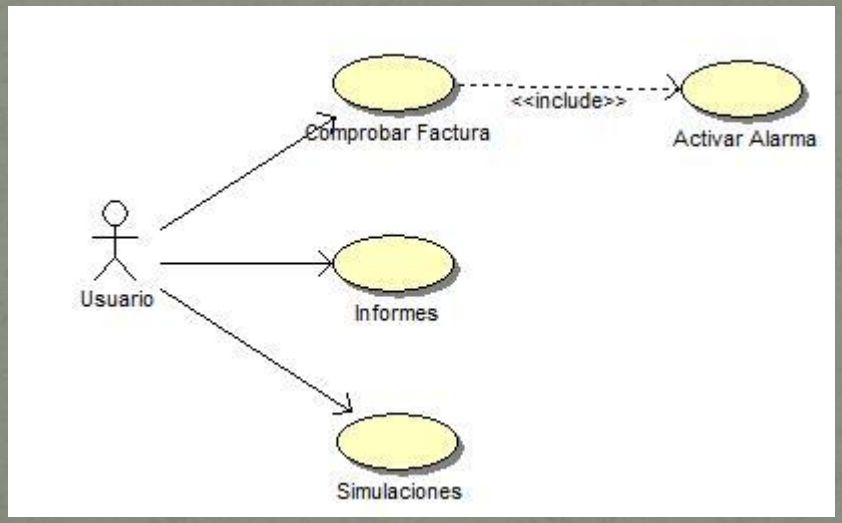
- Insertar Datos



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS DE LOS CASOS DE USO

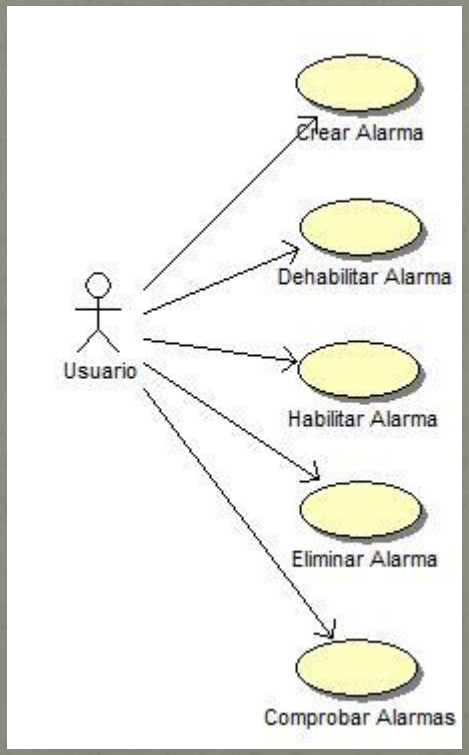
- Módulo de Consulta



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS DE LOS CASOS DE USO

- Alarmas



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

REQUISITOS NO FUNCIONALES

- Manejar gran volumen en consumos
- Interfaz intuitiva y amigable, usuarios inexpertos
- Normas de estilo página web del SNS-O
- Accesible desde plataformas de Microsoft
- Flexibilidad y adaptabilidad a los cambios

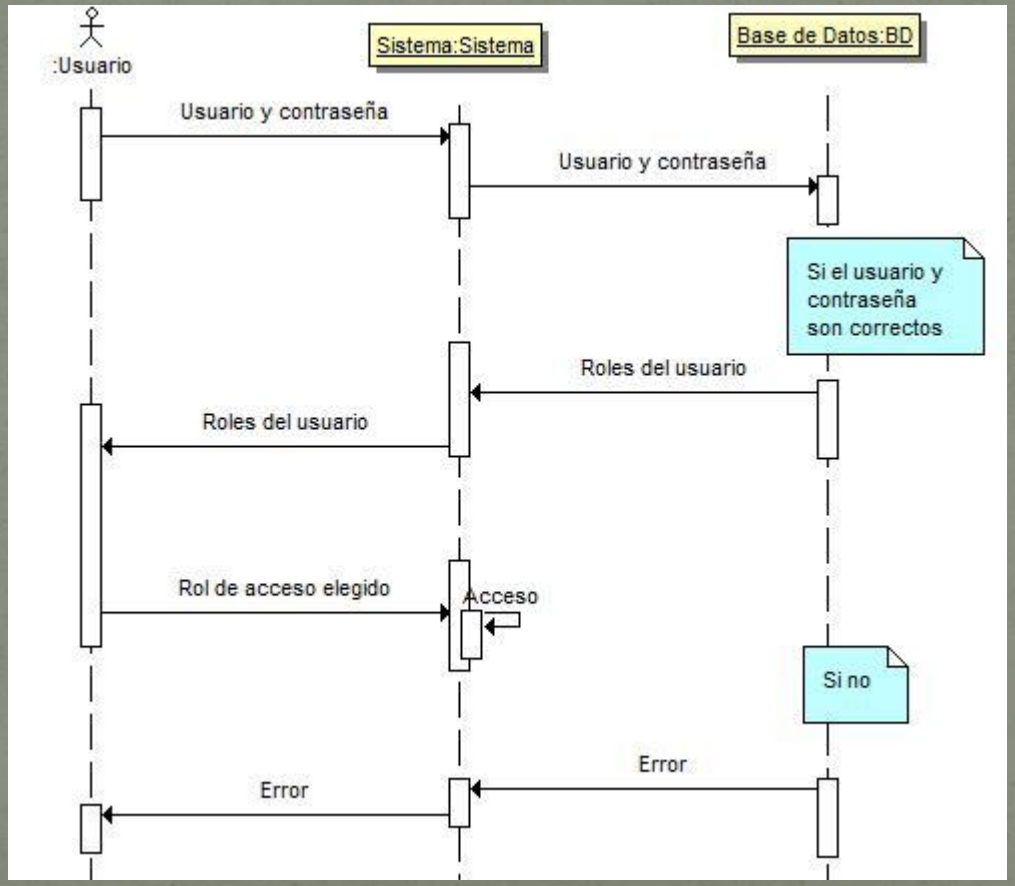
upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DISEÑO

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DISEÑO DE LOS CASOS DE USO

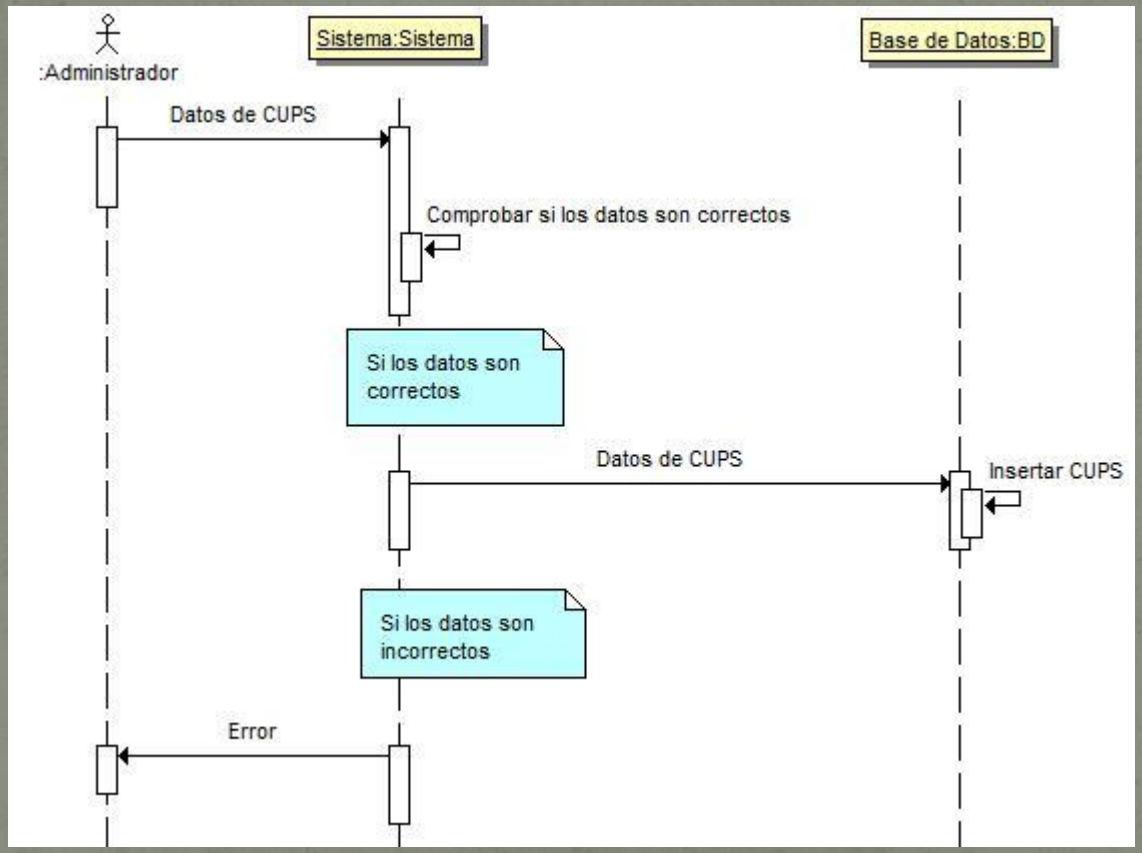
- Acceso al sistema



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DISEÑO DE LOS CASOS DE USO

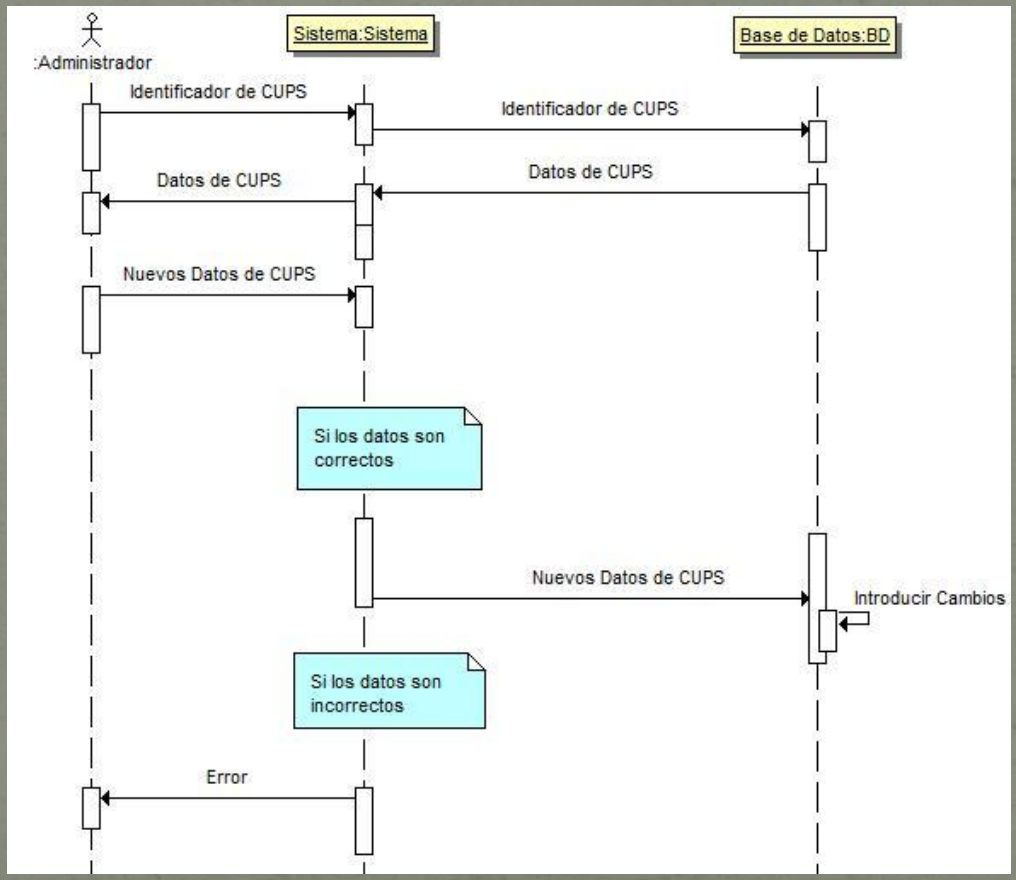
- Inserciones



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

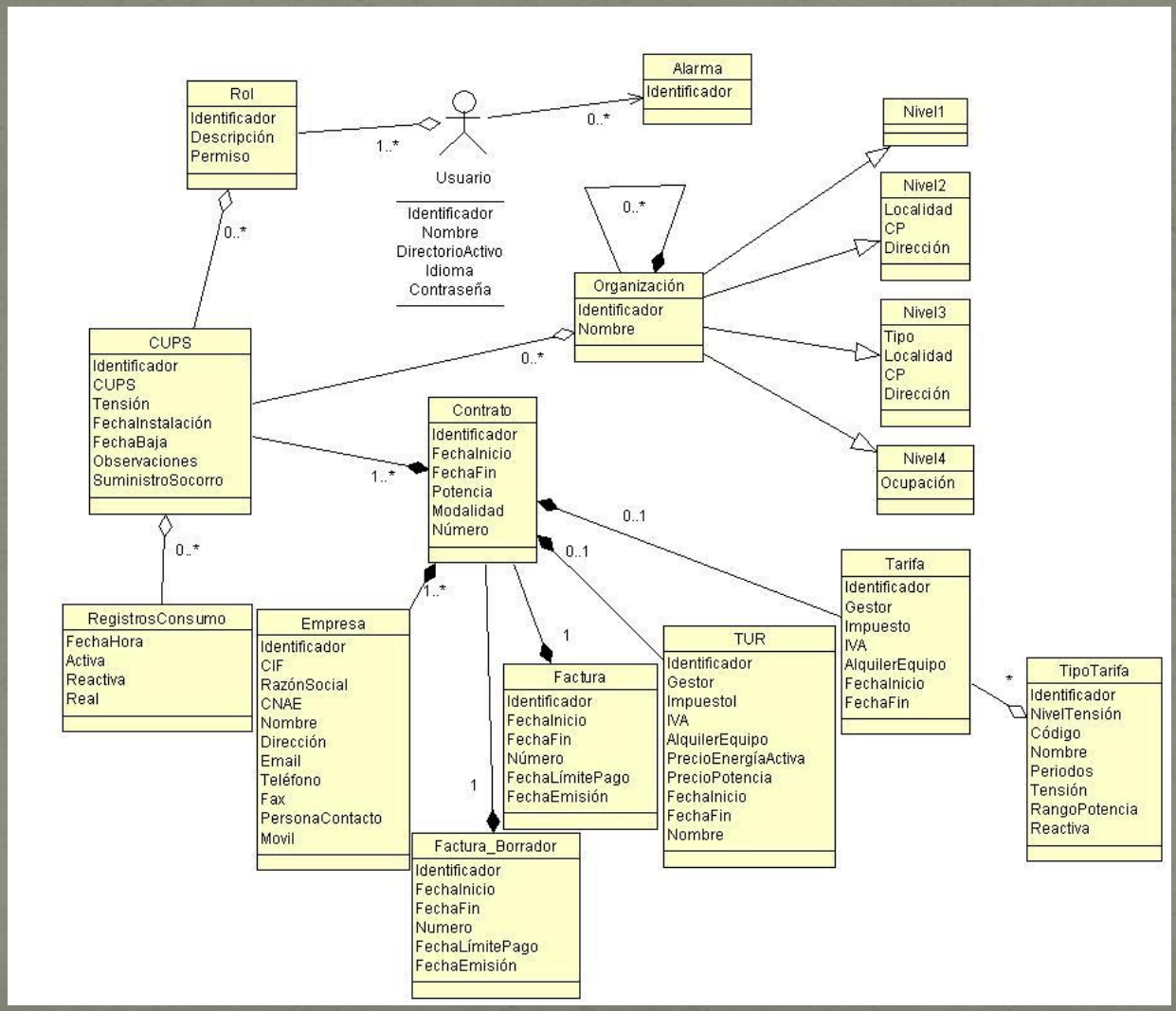
DISEÑO DE LOS CASOS DE USO

- Modificaciones



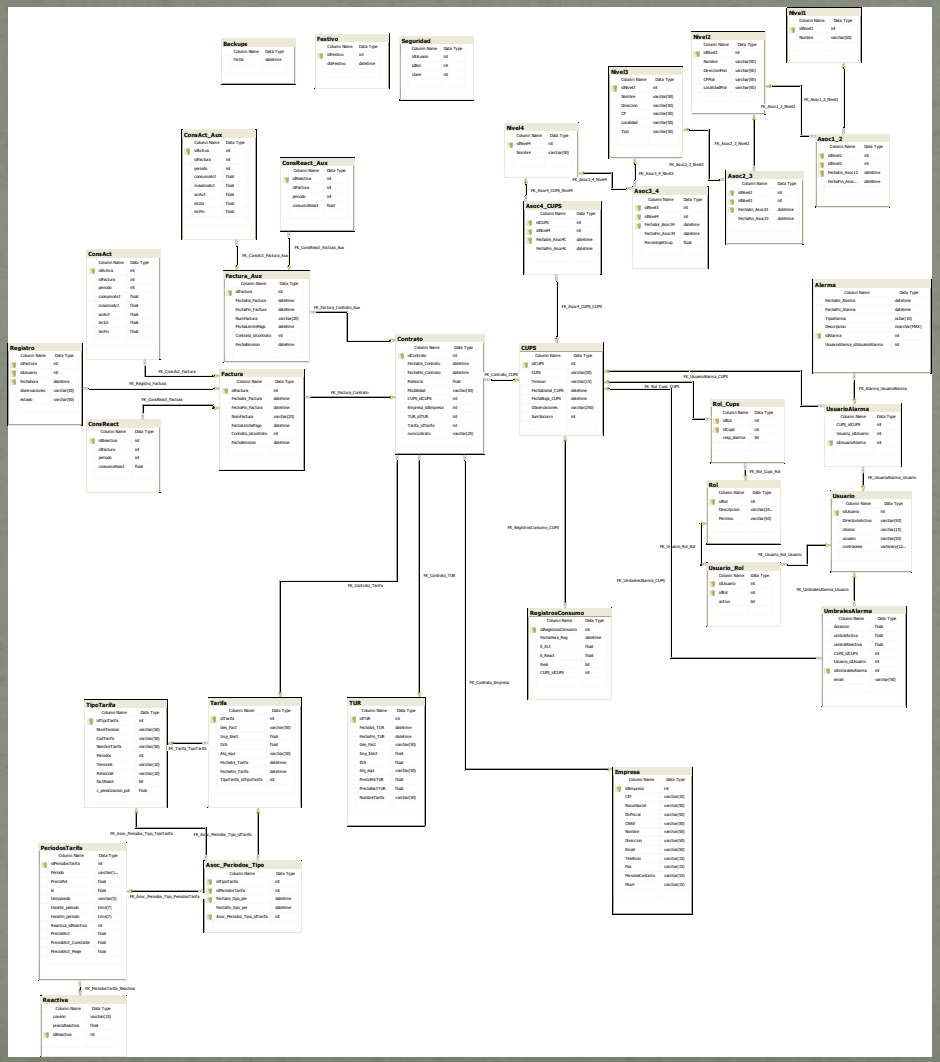
upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DIAGRAMA DE CLASES



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DIAGRAMA RELACIONAL DE LA BASE DE DATOS



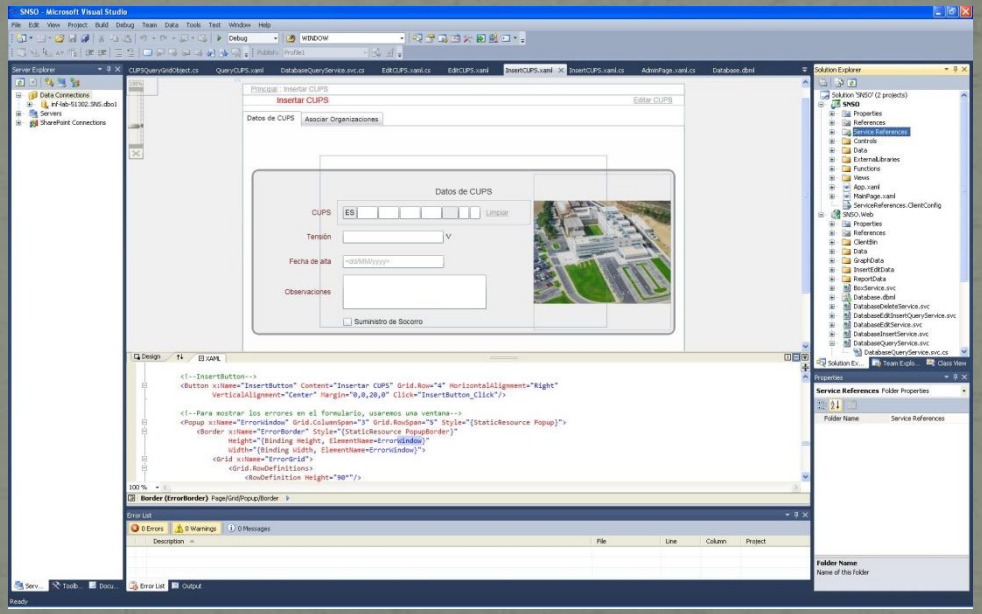
upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

IMPLEMENTACIÓN

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

PLATAFORMA DE DESARROLLO

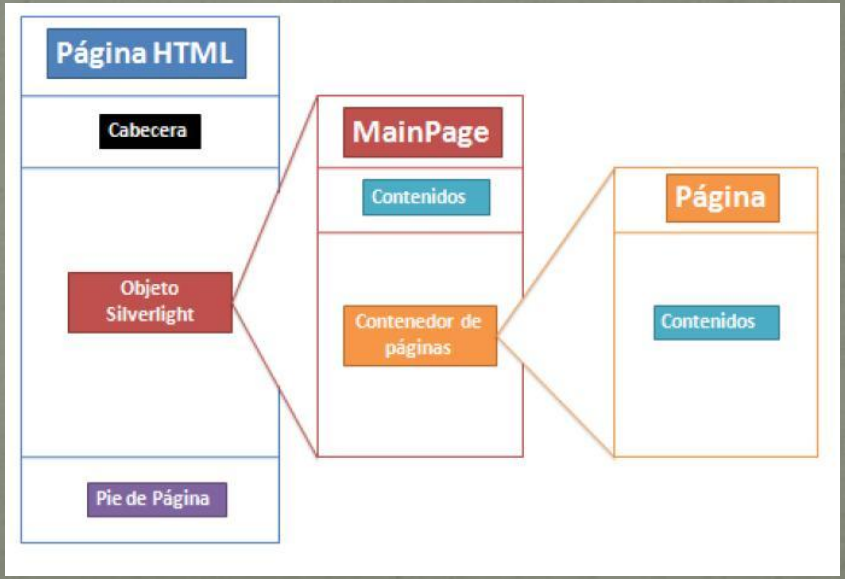
- Microsoft Visual Studio 2010
- SDK de Silverlight
- Toolkit de Silverlight



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ESTRUCTURA

- Página web HTML
- Contiene el objeto silverlight
- Página principal al estilo .NET
- Contiene las páginas



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONEXIÓN CON LA BASE DE DATOS

- Silverlight no puede conectarse directamente por seguridad
- Usa Servicios Web
  - Creación Modelo de Datos
  - Creación Servicios Web: conectándose a través del Modelo de Datos
  - Publicación de los servicios Web a través del IIS

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONCLUSIONES

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONCLUSIONES

- Dificultad al continuar un proyecto ya empezado
- Problemas con Endesa: La factura no coincide con los cálculos según el BOE.
- Problemas con Unión Fenosa: No presenta el mismo formato que Endesa
- Aprendizaje tecnologías de Microsoft.
- Aprendizaje de Silverlight y SQL Server.
- Trato con un cliente real.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

LÍNEAS FUTURAS

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

LINEAS FUTURAS

- Reestructuración de la BD y en el interfaz para adaptar la energía reactiva correctamente
- Mantener históricos en tarifas vs almacenar tarifas erróneas (superadministrador)
- Distribución informes entre usuarios
- Control facturas estimadas y abonadas
- Gráfica potencia consumida vs potencia contratada
- Alarmas automáticas

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

LINEAS FUTURAS

- Nuevos informes específicos de la facturación de Unión Fenosa, al igual que las simulaciones
- Otra aplicación similar, pero enfocado en el gas

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira